Applying Dynamic Bayesian Networks to Process Monitoring

by

Brandon Jason Wakefield

Thesis presented in partial fulfilment of the requirements for the degree of

Master of Engineering (Extractive Metallurgical Engineering)

in the Faculty of Engineering at Stellenbosch University

Supervisor:

Dr. R.S. Kroon

Computer Science Division
Stellenbosch University

Co-supervisor:

Dr. L. Auret

Department of Process Engineering
Stellenbosch University

2018

The financial assistance of the Centre for Artificial Intelligence (CAIR) and Stone Three Mining Solutions (S3MS) towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at are those of the author and are not necessarily to be attributed to CAIR or S3MS.

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: December 2018

Plagiarism Declaration

1. Plagiarism is the use of ideas, material and other intellectual property of another's work and to present is as my own.
2. I agree that plagiarism is a punishable offence because it constitutes theft.
3. I also understand that direct translations are plagiarism.
4. Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism.
5. I declare that the work contained in this assignment, except where otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.

Student number: 16975413

Initials and surname: B.J. Wakefield

Signature: 

Abstract

In efforts to reduce the impact of human error on the operation of chemical and mineral processing plants, reliable process monitoring solutions attempt to assist plant operators and engineers to detect and diagnose process faults before significant loss is incurred. An existing solution, the traditional multivariate statistical process monitoring (MSPM) approach, is able to reliably detect abnormal process behaviour but struggles to unambiguously identify the root cause of the abnormal behaviour. It was identified that this is caused by a lack of incorporation of existing process knowledge into the framework of the MSPM approach.

It was proposed to investigate a different fault diagnosis approach which directly incorporates process knowledge into its framework. Lerner *et al.* (2000) and Lerner (2002) present such an approach, using probabilistic methods to infer process behaviour given a particular process model. This model is in the form of a dynamic Bayesian network (DBN), and would contain various models which each describe particular process behaviour given information about the operational status of various process components. In particular, these DBN models were able to describe normal process behaviour in addition to highly specific abnormal process behaviour caused by, for instance, a sensor fault or a blocked pipe. Using optimised methods, the authors could then use a DBN model to make predictions about process behaviour and infer, given observation of actual process behaviour, which combination of component statuses best describe that observation. Therefore, solving the fault diagnosis problem could be reduced to performing inference in a DBN using this approach.

A probabilistic fault diagnosis (PD) approach based on Lerner *et al.* (2000) and Lerner (2002) was therefore implemented and investigated in this thesis. A survey of recent DBN-based PD approaches was also performed, and it was determined that relatively little research had been done on the topic. Furthermore, published results presenting fault diagnosis performance for DBN-based PD approaches were typically found to be useless for meaningful comparison with a traditional MSPM approach. In this regard, this thesis aimed to investigate the usefulness of the PD approach in comparison to the MSPM approach, while providing useful fault diagnosis performance metrics to facilitate comparison with other fault diagnosis approaches.

The PD approach tested in this research also extended upon Lerner *et al.* (2000)

and Lerner (2002) by including models for regulatory control systems and recycle streams based on the work by Yu and Rashid (2013). Additionally, from the same paper, the concept of abnormality likelihood index (ALI) was implemented in the PD approach. This enabled the PD approach to function more similarly to the MSPM approach, facilitating direct comparison.

Generally, it was found that the PD approach could provide competitive fault detection when compared with the MSPM approach. However, this was at the cost of real-time fault detection as well as longer detection delay for incipient faults. On the other hand, it was found that the PD approach performed better at root cause analysis than the MSPM approach. In particular, the PD approach typically provided better isolation for the root cause of fault conditions.

Despite some issues, similar results were observed for the PD approach when scaling up to larger processes. Nonetheless, these issues may be addressed with additional research, further improving the capabilities of the PD approach. Therefore, it was concluded that the PD approach is useful for fault diagnosis and should be investigated further in future research.

Abstrak

In pogings om die impak van menslike foute op die bedryf van chemiese en mineraalprosesseringsaanlegte te verminder, probeer betroubare prosesmonitering oplossings om aanlegoperateurs en -ingenieurs by te staan om prosesfoute op te spoor en te diagnoseer voor beduidende verliese aangegaan word. 'n Bestaande oplossing, die tradisionele meervariaat statistiese prosesmonitering (MSPM) benadering, kan abnormale prosesgedrag betroubaar opspoor maar sukkel om die grondoorsaak van die abnormale gedrag eenduidig te identifiseer. Dit is geïdentifiseer dat dit veroorsaak word deur 'n gebrek aan inkorporasie van bestaande proseskennis in die raamwerk van die MSPM-benadering.

Dis voorgestel om 'n ander foutdiagnose benadering te ondersoek, wat proseskennis direk in sy raamwerk inkorporeer. Lerner *et al.* (2000) en Lerner (2002) lewer so 'n benadering deur probabilistiese metodes te gebruik wat prosesgedrag aflei gegewe 'n bepaalde prosesmodel. Die model is in die vorm van 'n dinamiese Bayes-netwerk (DBN) en kan verskeie modelle bevat wat elk bepaalde prosesgedrag beskryf gegewe informasie oor die operasionele status van verskeie proseskomponente. Hierdie DBN modelle kon veral normale prosesgedrag bo-op hoogs gespesifiseerde, abnormale prosesgedrag veroorsaak deur byvoorbeeld, 'n sensorfout of geblokkeerde pyp, beskryf. Deur geoptimaliseerde metodes te gebruik, kan die outeurs dan 'n DBN model gebruik om voorspellings oor prosesgedrag te maak en aflei watter kombinasie van werklike komponent-statusse die observasie die beste beskryf, gegewe observasie van werklike prosesgedrag. Die oplossing van die foutdiagnose probleem kan dus gereduseer word tot die uitvoering van inferensie in 'n DBN, deur hierdie benadering te gebruik.

'n Probabilistiese foutdiagnose (PD) gebaseer op Lerner *et al.* (2000) en Lerner (2002) is daarom geïmplementeer en ondersoek in hierdie tesis. 'n Opname van onlangse DBN-gebaseerde PD-benaderings is ook uitgevoer, en dis vasgestel dat relatief min navorsing oor hierdie onderwerp gedoen is. Verder, gepubliseerde resultate wat foutdiagnose uitvoering vir DBN gebaseerde PD-benaderings wys, is tipies nutteloos gevind vir die vergelyking met 'n tradisionele MSPM-benadering. In hierdie opsig het die tesis beoog om die nuttigheid van die PD-benadering in vergelyking met die MSPM-benadering te ondersoek, terwyl dit nuttige foutdiagnose werkverrigtingmetrieke verskaf om die vergelyking met ander foutdiagnose-benaderings te fasiliteer.

Die PD-benadering wat in hierdie navorsing getoets is, het ook op Lerner *et al.* (2000) en Lerner (2002) uitgebrei deur modelle vir regulerende beheerstelsels en herwinningstrome in te sluit, gebaseer op die werk deur Yu and Rashid (2013). Daarby, uit dieselfde publikasie, is die konsep van abnormaliteit aanneemlikheidsindeks (AAI) geïmplementeer in die PD-benadering. Dit het die PD-benadering toegelaat om meer soortgelyk aan die MSPM-benadering te funksioneer, wat 'n direkte vergelyking fasiliteer.

Oor die algemeen is gevind dat die PD-benadering kompeterende foutdeteksie kon verskaf, as dit met die MSPM-benadering vergelyk word. Dit was egter ten koste van intydse foutdeteksie sowel as langer deteksie vertraging vir aanvangsfoute. Aan die ander kant is dit gevind dat die PD-benadering beter met grondoorsaak analise gedoen het as die MSPM-benadering. Die PD-benadering het in besonder tipies beter isolasie vir die grondoorsaak van foutkondisies verskaf.

Ten spyte van 'n paar kwessies, is soortgelyke resultate vir die PD-benadering waargeneem toe daar na groter prosesse opgeskaal is. Nietemin, hierdie kwessies kan geadresseer word met bykomende navorsing, wat die vermoëns van die PD-benadering verder sal verbeter. Die gevolgtrekking is dat die PD-benadering nuttig is vir foutdiagnose en moet verder ondersoek word in toekomstige navorsing.

Contents

Acronyms

2-TBN two-time-slice Bayesian network

aConn abrupt partial blockage fault
AE algebraic equation
ALI abnormality likelihood index used by the probabilistic fault diagnosis approach
AUC area under receiver operating characteristic curve

BN Bayesian network
BNT Bayesian Network Toolbox

CPD conditional probability distribution

DAEs differential and algebraic equations
DBN dynamic Bayesian network
DE differential equation

F1 score harmonic mean of precision and true alarm rate
FAR false alarm rate

iConn incipient partial blockage fault

JPD joint probability distribution

MSPM multivariate statistical process monitoring

N configuration with process and sensor noise
NC configuration with regulatory control as well as process and sensor noise
NCR configuration with recycle as well as regulatory control, and process and sensor noise
NOC normal operating conditions

P1s abrupt pressure sensor bias fault for sensor on tank 1
P2s abrupt pressure sensor bias fault for sensor on tank 2
PC principal component
PCA principal component analysis
PD probabilistic fault diagnosis

PDF probability density function
PI Proportional-Integral
PV process variable

ROC receiver operating characteristic
RV random variable

sN configuration with only sensor noise
SPE squared prediction error monitoring statistic used by the multivariate statistical process monitoring approach

TAR true alarm rate

Symbols

α_{nf}	conditional probability for transition from a normal to abnormal component status
$\mathcal{B}$	Bayesian network
$\boldsymbol{\beta}$	vector of weights for a linear Gaussian
c	hyper-parameter used by the probabilistic fault diagnosis approach to test similarity of transitions
$\tilde{c}_i$	diagnosis result reported by the multivariate statistical process monitoring approach and used to describe the i$^{\text{th}}$ process variable's relative contribution to abnormal behaviour
CV	controlled process variable
$_\Delta$	subscript used to denote discrete partition of a set of random variables
Δ_{bel}	random variable in a belief network, conditioned on H_{bel}, representing the set of discrete random variables $\mathbf{Q}_\Delta$
E	controller error
$_\Gamma$	subscript used to denote continuous partition of a set of random variables
Γ_{bel}	random variable in a belief network, conditioned on H_{bel}, representing the set of continuous random variables $\mathbf{Q}_\Gamma$
H_{bel}	random variable in a belief network used to represent hypotheses for the beliefs about a process state
J	similarity metric, based on average of Kullback-Leibler divergences, used by the probabilistic fault diagnosis approach
K	hyper-parameter used by the probabilistic fault diagnosis approach to control pool size of hypotheses
K_c	controller gain
m_{opmodes}	total number of operating modes defined by the dynamic Bayesian network model of a process
μ	mean

$\boldsymbol{\mu}$	mean vector
MV	manipulated process variable
$\mathcal{N}$	(conditional) Gaussian probability distribution
$\mathbf{O}$	observed random variables in the process state
$\Pr$	probability
$\Pr_T$	hyper-parameter used by the probabilistic fault diagnosis approach to intelligently limit the number of transitions enumerated during transition enumeration
$\mathbf{Q}$	unobserved random variables in the process state
$\mathcal{R}i$	set of random variables relevant to subsystem i
$\mathcal{R}i^+$	partition of the set of random variables associated with subsystem i which are defined as being inside that subsystem
$\mathcal{R}i^-$	partition of the set of random variables associated with subsystem i which are defined as being outside that subsystem
σ	standard deviation
σ^2	variance
$\boldsymbol{\Sigma}$	covariance matrix
SP	set point for controlled process variable
T_I	controller integral time constant
$T_{\max}$	hyper-parameter used by the probabilistic fault diagnosis approach to limit the absolute maximum number of transitions enumerated during transition enumeration
T^2	Hotelling's T^2 monitoring statistic used by the multivariate statistical process monitoring approach
w_i	weight of the i$^{\text{th}}$ Gaussian in a hybrid mixture of Gaussians

Chapter 1

Introduction

Human error is commonplace when performing manual tasks on chemical and mineral processing plants. These errors result from information overload due to distractions or inexperience, poor reasoning and judgement, or random error (Li *et al.*, 2011). In most industrial processes, regulatory control (such as the actuation of valves) is automated and does not require human operator intervention. However, abnormal events, such as the failure of an actuated valve, cannot be managed by regulatory control systems and often need to be manually diagnosed and resolved by operational staff.

Incorrect or late diagnosis of abnormal events may result in financial losses to the plant including unsafe operation, equipment damage, decreased production rate, and off-specification product quality. Automation of abnormal event management has become increasingly important as industrial trends have shifted toward using fewer employees with more responsibilities (Leopold *et al.*, 2016), increasing the need for reliable decision-making by a plant's operational staff. Reliable and efficient diagnostic tools for the operational staff may significantly reduce the frequency of human error occurring and/or reduce its impact when it does occur.

Abnormal event management concerns the timely detection of abnormal events (i.e. faults), the analysis of fault symptoms and their possible root causes, and fixing these root causes. This research is focused on the development of tools to aid abnormal event management, in particular the detection of faults and diagnosis of their possible root causes, through process monitoring. Process monitoring concerns the use of (real-time) process data to infer particular details about process behaviour — as an example, for diagnostic purposes, this data may be used to determine whether process behaviour is normal or abnormal.

This chapter serves as an introduction to process monitoring concepts and terminology, discusses the successes and shortcomings of previous research in the field of abnormal event management with respect to data-driven fault diagnosis, briefly introduces an alternative model-based approach for fault diagnosis, states the expected outcomes for this research, and presents a brief overview of the rest of the

thesis.

1.1 Aiding supervisory control with process monitoring

Almost all industrial processes have multiple layers of control (see Figure 1.1). Regulatory control allows automated operating point changes and the ability to minimise the impact of process disturbances. The objectives of regulatory control are strictly prioritised from safety to profit because every process variable (PV) is not individually controllable and the interconnected nature of most industrial processes imposes an upper limit to regulatory control (Marlin, 2000). Although multivariate regulatory control addresses interaction of PVs directly in the control system design, no regulatory control system would be able to completely remove variability in all PVs. These blind spots of regulatory control are mostly addressed by supervisory control which oversees the normal operation of regulatory control.

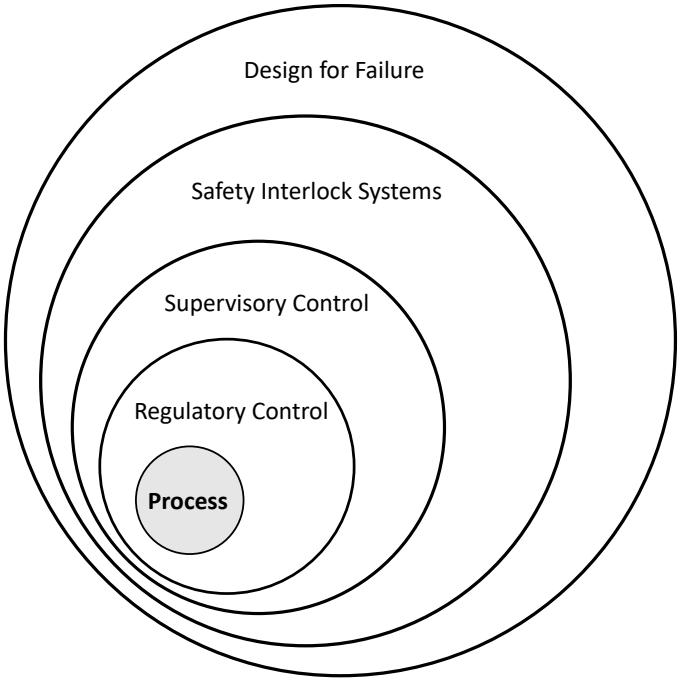


Figure 1.1: Layers of control in a typical industrial process.

Currently, most supervisory control is manual. It relies on humans making good judgements, given available process data and their own observations and experience, when analysing whether or not the process is operating safely and efficiently and if not, to identify and address the cause thereof. Supervisory control therefore extends beyond abnormal event management to include topics such as maintenance, operations training, health and safety, process optimization, and so forth. When

supervisory control works well, a process can operate without incident and near optimal efficiency for extended periods of time (Kaushik and Khanduja, 2009). However, the current manual approach to supervisory control in most chemical and mineral processing plants will always be subject to human error. Therefore, safety interlocks prevent incidents (both uncontrollable and those due to human error) from scaling catastrophically by automatically shutting down affected areas of a plant. The last layer of control, if all else fails, is to design for failure such that in the case of catastrophic failure, loss is contained and manageable.

The cost of using safety interlocks and design for failure is extremely high due to the need to replace equipment once activated in addition to operational downtime. Therefore, it is important to maintain control within the supervisory control layer. Ideally, fully automated supervisory control would provide a solution to this problem, but current research and technology is not advanced enough to provide such a solution (Bullemer and Reising, 2015; Dai *et al.*, 2016). Instead, current research employs the existing human input structure and attempts to develop tools to aid interpretation and decision making in the context of supervisory control. The development of these tools (such as those used for fault diagnosis) is the focus of process monitoring.

1.1.1 The fault diagnosis problem

Fault diagnosis is a two-part endeavour: the first part concerns timely detection of faults and the second part concerns accurate root cause analysis of fault symptoms. The aim of fault diagnosis is to determine if a process is in a *normal* or *faulty* operating mode and then, in the latter case, to reason about the cause of the *faulty* operating mode (Himmelblau, 1978).

A **fault** (or abnormal event) is any abnormal deviation from normal operating conditions (NOC). Assuming that the behaviour of a process changes according to its operating mode, a well-designed process should behave in accordance with its design specifications almost exactly when in the *normal* mode. Conversely, in the *faulty* mode its behaviour deviates from that under the normal operating mode and may result in unsafe and/or inefficient operation. The cause of fault conditions must be addressed as soon as possible because its impact (loss of safety, production, or money) grows the longer the fault persists.

Fault detection concerns the detection of abnormal events. This can be formulated as a binary classification problem for the operating mode of a process, i.e. *normal* or *faulty*. On most plants, timely and accurate detection of faults has the potential to significantly reduce the number of incidents and resulting loss. However, the task of fault detection generally suffers from a lack of meaningful process data since only certain PVs are measurable and only a fraction of these are actually measured on most plants. This is because it is expensive to buy and maintain reliable sensors especially beyond the immediate requirements of regulatory control. Therefore fault detection

methods must be robust to sparse measurement conditions and able to improve in data-rich operations.

Despite these challenges, fault detection is an easier problem than the identification of possible fault causes. The primary challenge when identifying the cause of a fault is to distinguish between the symptoms of the fault and the root cause of (or basic event causing) the fault, i.e. **root cause analysis**. For example, consider a pressure sensor on a fluid process stream producing abnormal readings: how does one determine if the sensor itself is faulty, if there is a leak in the pipe, or if an upstream pump is faulty? Manual root cause analysis could be applied on a trial and error basis, i.e. by servicing the suspected faulty components during a scheduled maintenance period. However, the manual trial and error approach may not always be reliable and it may be more practical to better identify the potential root causes of a fault from available process data, i.e. as part of an online fault diagnosis system.

The challenge with respect to using process data in this manner is that reasoning about the cause of fault symptoms (i.e. identified by abnormal deviation in one or more PVs) may lead to either the actual root causes of the fault, or more fault symptoms. In the latter case, this second set of fault symptoms is only related to, and not the actual root causes of, the fault — this is typically an issue for more complex processes. For instance, due to the highly interconnected nature of industrial processes, it is easy for a fault to start in one part of the process and propagate throughout most of the process before producing the first observed fault symptoms, especially when recycle streams are present in the process (Kelly and Lees, 1986).

It is also important to note that root cause analysis becomes more difficult in cases where a fault is subtle (such as for an incipient fault) where considerable time may pass, after the fault manifests, before fault symptoms are observed or before spectacular failure of a component occurs (such as in fatigue failure scenarios) (Himmelblau, 1978).

Fault diagnosis is the combination of fault detection and root cause analysis. Early approaches to fault diagnosis involved paper-based systems, such as hazard and operability (HAZOP) studies and safe operating procedures (SOPs), which relied on operators detecting faults and using lookup tables to identify possible causes (Lees, 2012; Bullemer and Reising, 2015). These approaches worked well for experienced operators with extensive knowledge of the processes they were running. However, when there is a lack of such operators, process monitoring tools may assist operators who have limited process knowledge to better diagnose abnormal events (Sheridan, 1981; De Keyser and Leonova, 2001; Yoshikawa and Zhang, 2014; Dai *et al.*, 2016).

1.1.2 Process monitoring and fault diagnosis

It may be possible to better perform fault diagnosis by presenting process data to operators in a different manner that is more effective for this task. Consider the simple heat exchanger shown in Figure 1.2, where temperature and flow measurements are available for all streams. Given that the sensors are working properly, it is possible to calculate an overall heat transfer coefficient for the unit and report the single value to the operator instead of, for example, four temperature and four flow measurements. When the overall heat transfer coefficient for the unit then drops below a certain threshold value, the operator should know to schedule maintenance for the unit after which it should operate as normal once again.

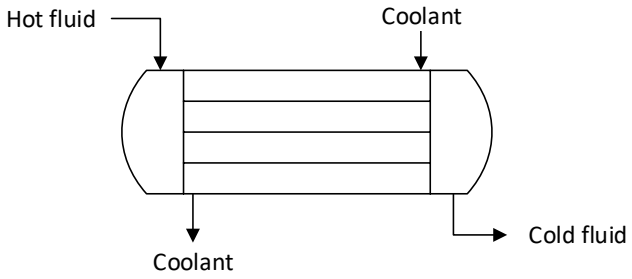


Figure 1.2: A simple heat exchanger.

This example illustrates a simple process monitoring solution which can provide an operator with more useful information to better detect and diagnose a heat exchanger fault. However, even in this simple example it is possible that many other aspects of the process may go wrong such as varying fluid compositions, varying fluid densities, valve or sensor faults, and so forth. Furthermore, in a typical process such a heat exchanger would not be in isolation, but be surrounded by other process units which may be just as susceptible to other faults. Therefore, it requires significant effort and experience from an operator to distinguish between relevant and irrelevant information, to identify fault conditions from the relevant information, and then isolate the cause of that fault.

Although auto-diagnostic systems are not yet available, it is possible to reduce information overload to operators and work toward an approach that could eventually become part of such an auto-diagnostic system. Ultimately, all process data may be directly fed into a process monitoring solution which may fit into the broader supervisory control scheme by alerting an operator only when human input is absolutely necessary (Reis and Gins, 2017). However, such a reliable and robust solution remains elusive, due to the highly complex and stochastic nature of industrial processes. Hence, current process monitoring solutions and tools provide a way

forward for improving fault diagnosis procedures by reducing the time required for diagnosis, improving fault detection accuracy, and providing better leads to the possible root causes of a fault.

Research in process monitoring for fault diagnosis is multidisciplinary and draws from disciplines such as statistics, machine learning, computer science, and engineering. Venkatasubramanian *et al.* (2003*a*,*b*,*c*) review a large selection of different approaches to tackling the problem of fault diagnosis. The authors view each approach as a series of transformations from raw measurements to a fault class related to the symptoms of the fault, as shown in Figure 1.3.

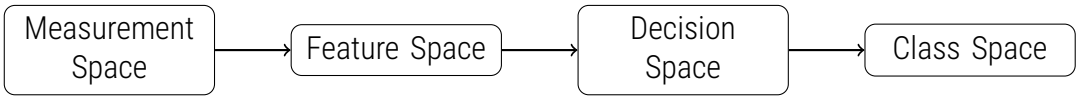


Figure 1.3: Transformations of data for fault diagnosis. Redrawn from Venkatasubramanian *et al.* (2003*b*).

Raw measurements are received from the process via sensors and are transformed into features. This transformation may involve removing measurements which do not contribute to fault diagnosis, combining multiple measurements into some single calculated value (such as in the heat exchanger example), or projecting raw measurements onto a feature space with fewer dimensions. Feature extraction thus facilitates a change in the number of measured PVs from the raw measurement space to the feature space, typically increasing the information content in each variable in the feature space relative to the information content in each variable in the raw measurement space. The next transformation, from the feature space to the decision space, computes some distance metric for a data sample describing how much the sample's features deviate from an expected distribution of features obtained from NOC. Finally, transformation from the decision space to the class space involves allocating a fault class (such as normal, sensor fault, actuator fault) to a sample based on the distance metric(s) computed during the transformation from the feature space to the decision space.

Venkatasubramanian *et al.* (2003*b*) identified that the first transformation from the raw measurement space to the feature space requires some sort of *a priori* process knowledge and the subsequent transformation to the decision space requires a suitable search strategy. This allowed the authors to classify vastly different fault diagnosis approaches according to their use of *a priori* process knowledge or search strategies.

The authors identify three varieties of *a priori* process knowledge, namely: quantitative model-based, qualitative model-based, and historical process data-based. Quantitative model-based knowledge is typically based on some fundamental or empirical

mathematical model of the process, whereas qualitative model-based knowledge consists of a series of rules (similar to those from HAZOP studies) which can be used to trace back to some root cause of a fault. Historical process data is sampled process data that has been collected over a long period of time, typically months. Most fault diagnosis approaches use one of these forms of *a priori* process knowledge to enable transformation from the raw measurement space to the feature space using some algorithm (such as principal component analysis).

Venkatasubramanian *et al.* (2003*a*) identified two broad classes of search strategies for fault diagnosis approaches: topographic and symptomatic. Topographic search identifies novelties in new data by comparison to a normal operation template. Symptomatic search instead uses fault symptoms to trace back to some root cause. However, the authors noted that the choice of search strategy is limited by the choice of *a priori* process knowledge since the *a priori* process knowledge determines the features available in the feature space. Therefore, the authors found it easier to classify fault diagnosis approaches according to their use of *a priori* process knowledge instead of search strategy.

Along with the available *a priori* process knowledge, these classifications are useful for determining which fault diagnosis approach is better suited for a particular process. However, in industry, simple solutions are often preferred due to the lower amount of effort required to develop and maintain them. This is why historical process data-based fault diagnosis approaches, especially those using principal component analysis (PCA), have seen a surge in popularity in recent years. Unfortunately, previous research (Wakefield *et al.*, 2018) found that while a simple PCA-based fault diagnosis approach may provide satisfactory fault detection, it may yield inconclusive root cause analysis due to the lack of incorporation of domain knowledge into the approach. On the other hand, model-based approaches may provide better root cause analysis at the cost of additional modelling requirements and increased fault detection latency (Venkatasubramanian *et al.*, 2003*b*).

1.2 Probabilistic fault diagnosis using dynamic Bayesian networks

In contrast to the traditional PCA-based multivariate statistical process monitoring (MSPM) approach to fault diagnosis presented in Kourti and MacGregor (1995), model-based approaches are designed using (empirical) process models and directly incorporate domain knowledge into the fault diagnosis framework. Model-based approaches typically attempt to explain abnormal process behaviour in terms of deviation of observed process behaviour from process model predictions.

Lerner *et al.* (2000) present such an approach which uses a probabilistic model

to predict process behaviour under various operating modes and infer the most likely operating mode given raw measurements. The hybrid dynamic Bayesian networks (DBNs) used in Lerner *et al.* (2000) are a type of probabilistic model that model temporal causal relationships between discrete and continuous random variables (RVs) using conditional probability distributions. Each discrete RV in the DBN represents the status of a component (sensor, valve, etc.) in the process and the continuous RVs typically represent a process variable (pressure, flow, temperature, etc.). Given a particular operating mode, the DBN model is able to generate probability distributions for the values of the PVs. Subsequently, the machinery of probability theory may be used to infer the relative likelihoods of the different operating modes given raw measurements. Fault conditions are then detected based on the relative likelihood of the normal and various faulty operating modes. In the event of a fault, its root cause may also be deduced from the relative likelihood of abnormal behaviour in each process component (i.e. by examining the probability distributions of the discrete RVs). Although this approach is computationally complex and expensive, Lerner *et al.* (2000) developed a special method (detailed in Chapter 3) to manage this complexity.

1.3 Research outcomes

1.3.1 Aims

This research investigates a DBN-based probabilistic fault diagnosis (PD) approach based on Lerner *et al.* (2000). Relatively few publications presenting DBN-based PD approaches exist (Lerner *et al.*, 2000; Lerner, 2002; Roychoudhury *et al.*, 2006, 2008, 2009, 2010; Yu and Rashid, 2013), most of which provide unusable results for comparison with a traditional MSPM approach. This thesis aims to address this shortcoming, by comparing a DBN-based PD approach to a traditional MSPM approach, in addition to serving as an introduction for process engineers to probabilistic techniques in a practical setting, i.e. fault diagnosis of processes.

Based on the results in Lerner *et al.* (2000), it is hypothesised that a DBN-based PD approach will provide more accurate root cause analysis at the cost of increased computational requirements when compared with a traditional MSPM approach.

1.3.2 Objectives

1. Dynamic modelling and simulation of example process units, including automated feedback control and the ability to simulate a variety of abnormal events.

2. Design and application of a DBN-based PD approach based on Lerner *et al.* (2000).

3. Benchmarking and comparison of fault diagnosis performance for the PD approach and a traditional MSPM approach for case studies of varying complexity.

1.3.3 Scope

a) This thesis is intended for an audience of graduate process engineers who have little familiarity with probability theory and process monitoring. It is assumed, however, that the audience is familiar with mathematical modelling, simulation and basic control of chemical and mineral processes.

b) Two case studies, inspired by Lerner *et al.* (2000), will be considered: a two-tank system and a five-tank system. Both case studies will feature different configurations of varying complexity. This includes the absence or presence of regulatory control systems as well as recycle streams.

c) It is intended to show proof of concept for the PD approach, thus further scalability of the approach will not be considered in this thesis.

d) The MSPM approach will be based on Kourti and MacGregor (1995) and will use PCA for feature extraction and diagnose fault conditions using the Hotelling's T^2 and squared prediction error (SPE) statistics.

1.3.4 Contributions

This thesis makes the following contributions:

- Introducing concepts in probabilistic techniques from a practical (engineering) perspective.
- Design, documentation, and testing of a DBN-based PD approach based on Lerner *et al.* (2000), including reporting fault diagnosis performance benchmark results for future comparison with other fault diagnosis approaches.
- Provision of software for the process simulations, DBN models, PD approach, and MSPM approach used in this thesis under an open-source license.
- Comparison of the PD approach with a traditional MSPM approach with respect to fault detection and root cause analysis.

1.4 Thesis overview

The thesis is organised in the following manner:

- **Chapter 2** presents a brief theoretical background. First it presents the concepts of the traditional MSPM approach to give the reader a basic understanding of a fault diagnosis approach. Then it presents existing approaches to measure fault diagnosis performance for any approach. Finally, it sets the stage for the PD approach by introducing basic concepts in probability theory and Bayesian networks.

- **Chapter 3** details a PD approach based on Lerner *et al.* (2000). It presents the approach on a conceptual level with more concrete examples provided in the appendices. It also details the modelling procedure used to create the DBN models used for process monitoring.

- **Chapter 4** presents an overview of recent developments in DBN-based PD approaches. A number of approaches are qualitatively compared with respect to the characteristics of a good fault diagnosis approach (Venkatasubramanian *et al.*, 2003*b*). Additionally, some concluding remarks are made about standardisation of testing results in future studies for better quantitative comparison between various fault diagnosis approaches.

- **Chapter 5** details the methods used for simulated data generation, the case studies considered for testing, the implementation of the PD and MSPM approaches, and the testing procedure used to determine the fault diagnosis performance.

- **Chapter 6** presents and discusses the fault diagnosis performance results for the PD approach as well as the effect of different hyper-parameter selections on these results. The performance of the PD approach is then compared with that of the MSPM approach. The chapter concludes with some caveats and a discussion of the limitations and some future directions of research for the PD approach.

- **Chapter 7** concludes the thesis with a brief summary of important findings and recommendations for future research.

Chapter 2

Theoretical Background

This chapter presents a collection of background material central to the rest of this thesis. The chapter begins with the presentation of a traditional MSPM fault diagnosis approach, covering both fault detection and root cause analysis. The MSPM approach showcases the basic mechanics of most data-driven fault diagnosis approaches. This is followed by a presentation of existing qualitative and quantitative methods for measuring the capabilities of a fault diagnosis approach for later comparison of the MSPM approach with the PD approach presented in Chapter 3. The chapter concludes with a brief introduction to Bayesian networks (BNs) in the context of fault diagnosis which serves as background for the PD approach.

2.1 Multivariate statistical process monitoring

The traditional MSPM fault diagnosis approach considered in this thesis is based on Kourti and MacGregor (1995). At its core, this approach uses historical process data to build a model describing process behaviour under NOC. Large deviations of raw measurements from the NOC model are considered abnormal[1] and the measured PVs which contribute the most to such an abnormal deviation are highlighted as fault symptoms. In this section, the MSPM approach is presented in detail. The section begins with a discussion of the characteristics of the historical process data needed to build the NOC model, then presents the methodology for feature extraction and the mechanism by which raw measurements are transformed to the feature space (see Figure 1.3). This is followed by the fault detection methodology, which includes the transformations from the feature space to the decision space and then the class space. Finally the root cause analysis methodology is presented.

2.1.1 Data characteristics

Typically, the MSPM approach requires one set of historical process data obtained under NOC to develop the NOC model. This data is typically structured as samples of

[1] Note that the changes in correlation structure of measured PVs are also considered abnormal.

the same measured PVs (which are typically continuous variables) at various points in time. A considerable amount of NOC data is typically required to build a model that adequately captures the variation of process data during normal operation while not wrongly classifying normal operation as faulty. The amount of NOC data required is an important limitation of the MSPM approach. Due to missing sensor readings, abnormal disturbances, faults, and shift changes it may be difficult to collect sufficiently large sets of NOC data. Furthermore, it is usually not possible to know if any historical process data collected was truly observed under NOC and thus free of faults.

In order to validate the diagnosis results of the MSPM approach, another set of NOC data is typically kept aside from the one used to develop the NOC model. This data set is used ensure that the MSPM approach correctly classifies samples from the additional data set as normal (not abnormal) operation. This additional data set may be obtained either by splitting a large NOC data set into two smaller data sets or by collecting another set of historical process data under NOC. The data set used to develop the NOC model is typically referred to as the training data, and the additional NOC data set is typically referred to as the validation data.

Lastly, to test fault diagnosis performance, most literature use process simulations to generate both normal and faulty data sets. The faulty data set is typically referred to as the test data and is used to benchmark the fault diagnosis performance of the approach either to show proof of concept or for comparison with alternative fault diagnosis approaches.

2.1.2 Feature extraction

Building a model of process variation from NOC data starts with feature extraction. PCA is a popular feature extraction technique which allocates the variance in a data set to a set of orthogonal principal component (PC) vectors (Jolliffe, 1986). Each PC vector defines a direction in the input space that the NOC data is projected onto. The PC vectors are the features of the NOC data set. Scores along each PC vector are obtained from projection of the NOC data onto the PC vectors. This enables variance in the NOC data set to be described by variances of the scores on each of the PC vectors and further enables the NOC data set to be described by a compact model (Venkatasubramanian *et al.*, 2003*c*).

It is common practice that each variable in the NOC data be standardised such that it has a mean of 0 and a standard deviation of 1. This will ensure that samples processed by the MSPM approach have the same scale and makes the approach more robust to outliers (Kourti and MacGregor, 1995). Let $\mathbf{X}_{\mathrm{NOC}}$ be an $m \times n$ matrix of m measurements of n PVs.[2] Let $\boldsymbol{\mu}_{\mathrm{NOC}}$ and $\boldsymbol{\sigma}_{\mathrm{NOC}}$ each be a $1 \times n$ vector of means and standard deviations respectively, such that $\mu_{i,\mathrm{NOC}}$ and $\sigma_{i,\mathrm{NOC}}$ respectively represent

[2]Note that m and $\mathrm{rank}\,(\mathbf{X}_{\mathrm{NOC}})$ are assumed to be larger than n.

the mean and standard deviation of each $m \times 1$ column, $\mathbf{x}_{i,\mathrm{NOC}}$, of $\mathbf{X}_{\mathrm{NOC}}$. Each variable $i \in \{1, \ldots, n\}$, $\mathbf{x}^s_{i,\mathrm{NOC}}$ is the i-th $m \times 1$ column of the standardised data set, $\mathbf{X}^s_{\mathrm{NOC}}$, obtained from

$$\mathbf{x}^s_{i,\mathrm{NOC}} = \frac{\mathbf{x}_{i,\mathrm{NOC}} - \mu_{i,\mathrm{NOC}}}{\sigma_{i,\mathrm{NOC}}}. \tag{2.1}$$

The $n \times n$ empirical covariance matrix of $\mathbf{X}^s_{\mathrm{NOC}}$, $\mathbf{\Sigma}_{\mathrm{NOC}}$, is defined as

$$\mathbf{\Sigma}_{\mathrm{NOC}} = \frac{\left(\mathbf{X}^s_{\mathrm{NOC}}\right)^{\mathrm{T}} \mathbf{X}^s_{\mathrm{NOC}}}{m-1}. \tag{2.2}$$

The directions of the PC vectors can be obtained by eigenvalue decomposition of $\mathbf{\Sigma}_{\mathrm{NOC}}$. The eigenvalue decomposition of $\mathbf{\Sigma}_{\mathrm{NOC}}$ into $\mathbf{P}$ and $\mathbf{\Lambda}$ is described by Equation 2.3. $\mathbf{P}$ is the desired $n \times n$ matrix of PC vectors such that each column in $\mathbf{P}$ describes the direction of one PC. $\mathbf{\Lambda}$ is an $n \times n$ diagonal matrix for which the vector $\boldsymbol{\lambda}$ represents the entries along the diagonal sorted in descending order, these are the eigenvalues of $\mathbf{\Sigma}_{\mathrm{NOC}}$.

$$\mathbf{P}\mathbf{\Lambda}\mathbf{P}^{-1} = \mathbf{\Sigma}_{\mathrm{NOC}} \tag{2.3}$$

Projection of the data in $\mathbf{X}^s_{\mathrm{NOC}}$ onto the columns of $\mathbf{P}$ yields an $m \times n$ matrix of scores, $\mathbf{T}_{\mathrm{NOC}}$, that is obtained from

$$\mathbf{T}_{\mathrm{NOC}} = \mathbf{X}^s_{\mathrm{NOC}}\mathbf{P}. \tag{2.4}$$

MATLAB® has a `pca` function that performs all of the operations from Equation 2.1 through to Equation 2.4 and simplifies implementation of the MSPM approach in MATLAB® .

The projections of the data onto each PC captures the variance of the data in the direction of that PC. Each PC explains a portion of the variance in the data along its direction and each entry in $\boldsymbol{\lambda}$ is proportional to the variance explained by the corresponding PC. Normalising $\boldsymbol{\lambda}$ thus yields the fractions of variance explained by each PC in descending order.

Typically, most of the variance in $\mathbf{X}^s_{\mathrm{NOC}}$ is explained by a small fraction of PCs. Choosing to retain only the first a PCs, which cumulatively explain a reasonable percentage of the variance in $\mathbf{X}^s_{\mathrm{NOC}}$, typically enables dimensionality reduction of the extracted features without significant degradation of the information originally contained in $\mathbf{X}^s_{\mathrm{NOC}}$ (Dong and McAvoy, 1996).

Let $\mathbf{P}_a$ be an $n \times a$ matrix of a retained PCs, i.e. the first a columns of $\mathbf{P}$. Let $\mathbf{T}_{a,\mathrm{NOC}}$ be the corresponding $m \times a$ matrix of scores for each of the PCs in $\mathbf{P}_a$. $\mathbf{X}^s_{\mathrm{NOC}}$ may then be approximately reconstructed as $\hat{\mathbf{X}}^s_{\mathrm{NOC}}$ by the inverse of the projection in Equation 2.4, instead using $\mathbf{P}_a$ and $\mathbf{T}_{a,\mathrm{NOC}}$ as shown in Equation 2.5.

$$\hat{\mathbf{X}}^s_{\mathrm{NOC}} = \mathbf{T}_{a,\mathrm{NOC}} \left(\mathbf{P}_a\right)^{\mathrm{T}} \tag{2.5}$$

2.1.3 Fault detection

Fault detection is enabled by comparing the scores obtained from feature extraction of standardised NOC data to the scores obtained from feature extraction of new standardised data samples. This section describes the procedure for obtaining the scores for the new standardised data samples and subsequently using them to perform fault detection.

Let $\mathbf{x}$ be a sample of size $1 \times n$ containing one value for each PV. To ensure that the sample has the same origin and scale as the NOC data, it is important to standardise each variable $i \in \{1, \ldots, n\}$ in the sample using the mean and standard deviation of the NOC data, i.e. $\mu_{i,\mathrm{NOC}}$ and $\sigma_{i,\mathrm{NOC}}$ respectively, for each variable. Then each entry in the standardised sample $\mathbf{x}^s$ is given by

$$x_i^s = \frac{x_i - \mu_{i,\mathrm{NOC}}}{\sigma_{i,\mathrm{NOC}}}. \tag{2.6}$$

The sample's scores, $\mathbf{t}$ of size $1 \times a$, may be obtained from $\mathbf{x}^s$ by projection onto the retained (from NOC data) PCs $\mathbf{P}_a$ according to Equation 2.7.

$$\mathbf{t} = \mathbf{x}^s \mathbf{P}_a \tag{2.7}$$

Comparison of $\mathbf{t}$ and $\mathbf{T}_{a,\mathrm{NOC}}$ may be visualised by plotting them on a scatter plot representing the feature space defined by $\mathbf{P}_a$ (and perhaps viewing only 2 or 3 dimensions at a time). This approach enables qualitative comparison of new sample scores ($\mathbf{t}$) to NOC scores ($\mathbf{T}_{a,\mathrm{NOC}}$).

Kourti and MacGregor (1995) present an alternative quantitative approach for fault detection based on two monitoring statistics, namely: (1) a modified Hotelling's statistic (T^2) and (2) the squared prediction error statistic (SPE). These are discussed briefly in the ensuing subsections.

Fault detection using modified Hotelling's statistic

Let $\boldsymbol{\lambda}_a$ be the largest a eigenvalues in $\boldsymbol{\lambda}$. The T^2 value for $\mathbf{t}$, defined by Equation 2.8, describes the variance-weighted distance of $\mathbf{t}$ from the origin of the feature space.

$$T^2 = \sum_{j=1}^{a} \frac{t_j^2}{\lambda_j}. \tag{2.8}$$

Fault conditions may be detected by monitoring the T^2 statistic and setting a threshold, T^2_{limit}, beyond which deviation from NOC is defined as abnormal. In other words, when a sample produces a $T^2 > T^2_{\text{limit}}$, its scores are considered significantly different from the scores obtained under NOC.

Fault detection using squared prediction error statistic

Recall that features are extracted once-off from a set of NOC data by performing eigenvalue decomposition of its covariance matrix (see Equation 2.3). Since new data is continuously generated during the operation of a process, it is possible for the new data to have a different covariance matrix to the original NOC data set. When this happens, eigenvalue decomposition of the new data set produces PCs and eigenvalues which are different from those produced from the original NOC data set. This means that the axes of the feature space for the new data set are different from those of the original NOC data set. Because of this, there is a risk that information may be lost when projecting a sample onto the retained PCs of the original NOC data set. Information loss can be monitored by comparing the approximate reconstruction of the sample, $\hat{\mathbf{x}}^s$, as shown in Equation 2.9 to the sample, $\mathbf{x}^s$, itself.

$$\hat{\mathbf{x}}^s = \mathbf{t}\left(\mathbf{P}_a\right)^{\mathrm{T}} \tag{2.9}$$

Since $\hat{\mathbf{x}}^s$ should closely match $\mathbf{x}^s$, the differences between the entries in the two vectors can be seen as information loss as result of projection onto the retained PCs of the original NOC data set. The SPE statistic summarises information loss for a sample into a single number, i.e. the sum of the squared residuals between $\mathbf{x}^s$ and $\hat{\mathbf{x}}^s$ as shown in Equation 2.10.

$$\mathrm{SPE} = \sum_{i=1}^{n} \left(x_i^s - \hat{x}_i^s\right)^2 \tag{2.10}$$

Therefore, if $\mathrm{SPE}_{\mathrm{limit}}$ is the threshold value for determining whether a sample represents normal or fault conditions, then when $\mathrm{SPE} > \mathrm{SPE}_{\mathrm{limit}}$ it suggests that a change in the correlation structure of the monitored variables has occurred. In other words, the process model (describing NOC) is no longer valid and thus the features extracted from NOC data no longer reliably represent the features of current process data.

Figure 2.1 shows a graphical representation of fault detection using the two monitoring statistics for two retained PCs. P_1 and P_2 create a two dimensional space (i.e. a plane) upon which the T^2 statistic operates. The remaining axis, upon which the SPE statistic operates, is a lumped representation of the features which differ from the retained ones and includes all the discarded features.

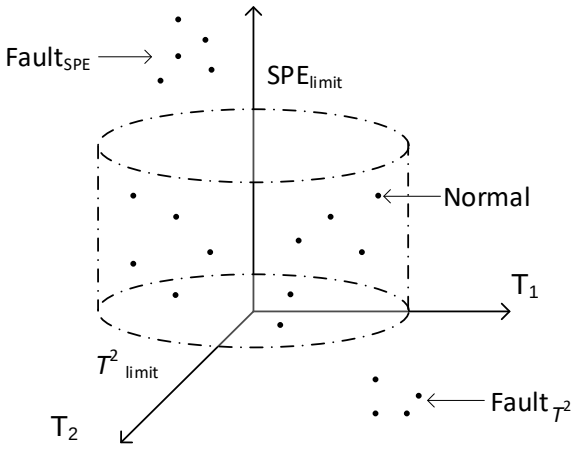


Figure 2.1: MSPM approach for fault detection. Limits for the modified Hotelling's (T^2) and squared prediction error (SPE) monitoring statistics describe an elliptical prism (for two retained PCs in the feature space) beyond which samples are classified as representing fault conditions.

Fault conditions are detected when monitoring statistics of a sample exceed either the T^2_{limit} or SPE$_{\text{limit}}$. Therefore, the limits place a bound on the monitoring statistics beyond which a sample can no longer be considered as representing NOC. In practice, these limits should be chosen carefully for optimal and reliable detection of fault conditions. Section 2.2 discusses various metrics which may be used to quantify fault detection performance, enabling the MSPM approach to be tuned by off-line analysis.

2.1.4 Root cause analysis

Kourti and MacGregor (1995) also describe a method for determining which PVs contributed most to the detection of fault conditions. Recall from Equation 2.10 that SPE is the sum of squared residual errors for a sample. Therefore, the contribution, c_i, of any variable, i, in that sample to the samples total SPE value is simply that variable's squared residual error, i.e.

$$c_i = (x_i - \hat{x}_i)^2 . \tag{2.11}$$

Typically, for any sample, there are differences between $\hat{\mathbf{x}}^s$ and $\mathbf{x}^s$ even under NOC due to the dismissal of PCs. Therefore, each variable has a nominal contribution to SPE under NOC, $\bar{c}_{i,\text{NOC}}$, which may be calculated by taking the average of that variable's contributions to SPE under NOC (i.e. using the training data set). The relative contribution score (Aldrich and Auret, 2013) for a variable i in a sample, $\tilde{c}_i$, accounts for

that variable's nominal contribution under NOC by considering c_i as a portion of $\bar{c}_{i,NOC}$:

$$\tilde{c}_i \equiv \frac{c_i}{\bar{c}_{i,NOC}}. \tag{2.12}$$

Relative contribution scores enable easier identification of variables which contribute more to detection of fault conditions in a sample. If fault conditions are reported for multiple samples, the relative contributions for these samples can be averaged for each variable and used to determine which variables are more likely to be associated with the detection of fault conditions. More specifically, if a variable's (average) relative contribution is greater than one, then that variable contributed more than its average nominal amount to the detection of fault conditions and should be investigated further, as illustrated in Figure 2.2.[3]

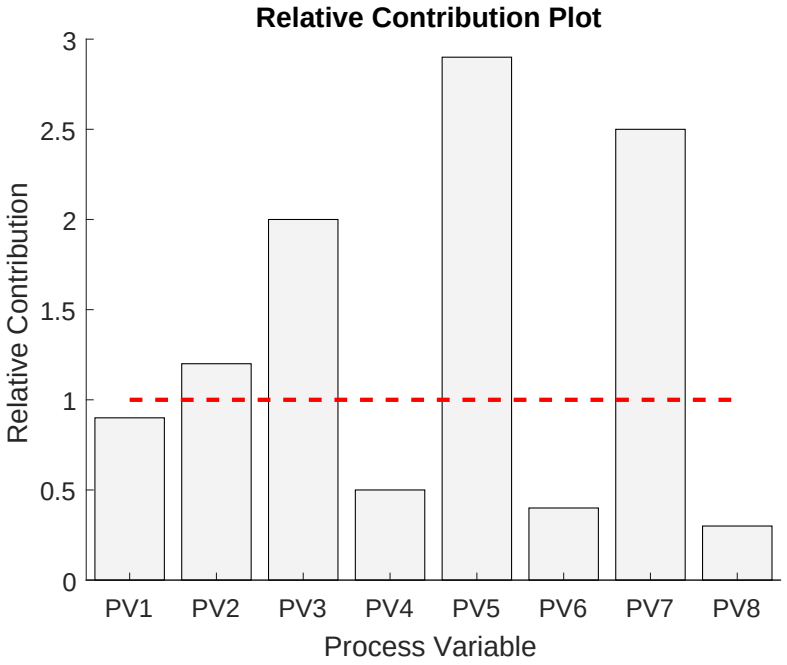


Figure 2.2: Illustration of a relative contribution plot. Here, the PVs (PV) 2, 3, 5, and 7 contribute more than their nominal amount to the detection of fault conditions and should be investigated further.

It is important to note that although the variables with high relative contribution act as indicators toward the cause of a fault, they may not necessarily point toward its root cause. For instance the symptoms of a fault, such as a stuck valve, in one area of a process where there are no sensors may propagate to a downstream area in the

[3] Note that Wakefield *et al.* (2018) use contributions relative to the 99th percentile of nominal amounts. This is an alternative use of relative contributions which places more emphasis on contributions higher than 99th percentile maximal values.

process where there are sensors, some of which may then be identified as having higher relative contribution to detection of the fault condition. Although these sensors may be associated with the fault conditions, the actual root cause is the stuck valve in the upstream process area. Therefore, the relative contributions can only identify indicators for symptoms of fault conditions from which expert and process knowledge (Maurya *et al.*, 2004) or data-driven methods (Lindner and Auret, 2015) then need to be used to determine the root cause of the fault.

2.2 Benchmarking fault diagnosis capability

Venkatasubramanian *et al.* (2003*b*) present various criteria for benchmarking the fault diagnosis capability of different fault diagnosis approaches. These criteria enable comparison of different approaches with one another as well as comparison of different model development and threshold selection for the same approach. This section presents a summary of both qualitative and quantitative criteria to analyse fault diagnosis capability for various fault diagnosis approaches.

First, the characteristics of a good fault diagnosis approach are presented. These characteristics enable qualitative[4] evaluation of a fault diagnosis approach based only on a detailed description of the approach, i.e. without requiring implementation. In addition to facilitating comparison of different approaches, these characteristics may also be used to guide the development of new fault diagnosis approaches. Next, the use of simulated process data for controlled testing of fault diagnosis approaches is discussed. The use of simulated process data facilitates post-implementation analysis of fault diagnosis capability for an approach under ideal and non-ideal scenarios. In particular, quantitative metrics are typically used in literature to evaluate fault detection performance while the performance of root cause analysis tends to be qualitatively evaluated. The evaluation of fault detection and root cause analysis performance concludes this section.

2.2.1 Characteristics of a good fault diagnosis approach

Ideally, a good fault diagnosis approach should possess each of the following characteristics (Venkatasubramanian *et al.*, 2003*b*):

1. Quick detection and diagnosis
2. Good isolability
3. High robustness
4. Good novelty identifiability

[4] Note the criteria may also be quantified and are not necessarily only qualitative.

5. Good estimation of classification error
6. High adaptability
7. Good explanation facility
8. Low modelling requirements
9. Reasonable storage and computational requirements
10. Good multiple fault identifiability

Quick detection and diagnosis refers to the sensitivity of the approach to true fault conditions and the detection latency of the approach. A highly sensitive fault diagnosis approach would often produce some false positives. This occurs when the approach reports that a fault is occurring, but in reality one is not. It is desirable to have a fault diagnosis approach that correctly detects fault conditions as soon as a fault occurs. However, low false positives and high true positives are competing objectives since the more sensitive a fault diagnosis approach is, the more susceptible to noise it becomes.

Good isolability refers to the ability of an approach to distinguish between different root causes (basic events) of a fault under ideal conditions, i.e. no noise and no modelling uncertainty. This ability relies heavily on the process knowledge embedded in the process model used by the approach and the degree to which the approach relies on its predictions. Typically, an approach with an accurate process model relies heavily on the predictions of its model and is able to isolate faults well. However, it may fail to reject the predictions of its model under conditions when that model becomes invalid, i.e. when the model becomes inaccurate due to noise (uncertainty), process disturbances, or operating point changes.

High robustness refers to the ability of an approach to perform well despite the presence of noise and modelling uncertainties. It puts an emphasis on consistently good performance in diverse circumstances rather than good performance only under ideal conditions. Therefore, a trade-off exists between having good isolability and high robustness.

Good novelty identifiability is desirable for a fault diagnosis approach because the full set of faults that may manifest in a process is seldom known. Typically, faults that may manifest in a process are captured from process knowledge and data. Incomplete process knowledge results in modelling uncertainties and partial knowledge of the full set of basic events that may manifest in a process. This can cause faults with unknown basic events to not be detected or to be incorrectly diagnosed.

Good estimation of classification error refers to the ability of an approach to provide an *a priori* indication of the confidence it has when determining whether or not a process is in a normal or faulty operating mode. A good estimate of classification error in a fault diagnosis approach enables users to better understand the reliability of its subsequent diagnosis results.

High adaptability refers to the ability of a fault diagnosis approach to have its process-specific intricacies (such as its process model) changed easily. Since processes evolve due to changes in raw feed materials or structural changes, it is desirable to be able to easily update the model embedded in a fault diagnosis approach to reflect these changes.

Good explanation facility refers to the ability of an approach to explain the origin and propagation of a basic event throughout a process. Similar to good estimation of classification error, this characteristic enables users to better understand the reliability of root cause analysis results.

Low modelling requirements refers to the amount of *a priori* knowledge required to create a fault diagnosis approach for a particular process. Ideally this should be minimal, causing a potential trade-off with good isolability.

Reasonable storage and computational requirements enable a good fault diagnosis approach to run in real-time on commercially available hardware.

Good multiple fault identifiability refers to the ability of an approach to detect and distinguish between the occurrence of two or more simultaneous faults (with different root causes).

Although no fault diagnosis approach possesses all of the above characteristics, most of these characteristics are qualitative scales and can still be used as a basis for evaluating and comparing fault diagnosis approaches.

2.2.2 Process modelling for simulated data generation

For quantitative evaluation of fault diagnosis capability, most fault diagnosis approaches are developed (trained) and evaluated using data generated from simulations of processes. Process simulations provide more control (versus a physical plant) over process behaviour by allowing the user to manipulate process disturbances and the occurrence of abnormal events. This enables the user to generate process data under known conditions with known abnormal events and use the data to evaluate the capability of an approach to reliably and accurately diagnose a fault.

When creating a process model for simulated data generation one typically begins

with a mathematical process model which represents normal operation of the process and then adds fault models representing abnormal events which can be triggered and controlled by the user as per their requirements for the generated data. The procedures for creating such a model are discussed in the ensuing subsections and an example application thereof is made available for a simple two-tank system in Appendix B.1. This same two-tank system is used to generate data for measuring the fault diagnosis performances of the approaches considered in Chapter 6.

2.2.2.1 Modelling for normal operating conditions

Chemical and mineral processing operations are often complex, consisting of many smaller process units which interact with one another to convert raw materials (inputs) into products and waste (outputs). To illustrate, a generic black-box process is shown in Figure 2.3.

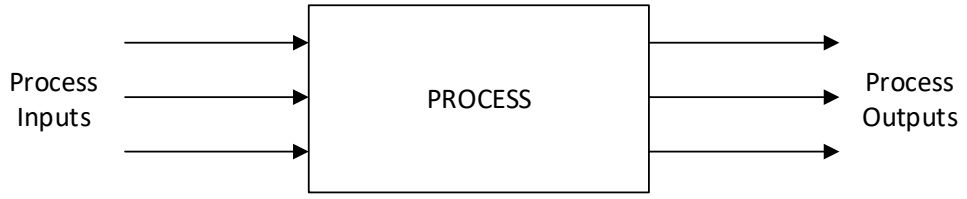


Figure 2.3: A generic process which transforms input (raw materials) streams into output (products and waste) streams.

The goal of open-loop process modelling is to determine, to a particular degree of granularity, the relationships between the inputs and outputs of a process in the absence of regulatory control. Empirically these relationships may be determined by perturbing some of the process inputs and observing the response of the process outputs.

For the purpose of developing a regulatory process control system, determining these relationships as input-output transfer functions in the Laplace domain may be sufficient. In order to design and implement a typical feedback process control system, it is only required that there exist manipulatable inputs (such as control valves) which are sufficiently correlated with the measured outputs of the process (Marlin, 2000). However, simple Laplace domain transfer function models are not sufficient to generate data for benchmarking fault diagnosis capability. The key reasons for this are because transfer functions models are linear and use lumped parameters. The former means that the model may no longer be valid when process inputs change, typically due to operating point changes, this causes process behaviour to differ from that at the initial point of linearisation. The latter means that many fundamental model parameters, a few of which may change under fault conditions, are typically lumped

into transfer function constants and the effect of a fault causing changes in a few of these parameters typically has unknown effects on the transfer function constants when a fundamental model is not available.

The laws of physics and thermodynamics as well as domain-theoretic knowledge, such as that of fluid mechanics and mineral processing, and semi-empirical relations are typically used to develop fundamental process models. Note that the following outline of fundamental process modelling is highly condensed and for more detailed treatments thereof, the interested reader is referred to Marlin (2000), Isermann (2006) and Perry and Green (2007).

A fundamental process model is a set of differential and algebraic equations (DAEs) that describes the relationships between the inputs and outputs of a process in the absence of regulatory control. It is typically built from the ground up, i.e. using deductive reasoning, in an attempt to explain physical phenomena. This requires significantly more knowledge and effort to develop than an empirical model since fundamental model dynamics are typically more complex and they require that the theory of the physical phenomena causing the dynamics be well understood. Fundamental process models typically consist of three types of equations: governing equations, constitutive equations, and constraints.

The governing equations are the conservation of mass (Equation 2.13) and the conservation of energy (Equation 2.14) defined over a particular control volume – such as for an individual unit process which has various process streams entering and exiting that unit. These governing equations are typically differential equations (DEs). Note that governing equations for the various unit processes may be combined with one another to obtain the governing equations for an entire process.

$$\begin{aligned} \{\text{accumulation of mass}\} &= \{\text{mass in}\} - \{\text{mass out}\} \\ \frac{\mathrm{d}m}{\mathrm{d}t} &= \dot{m}_{\text{in}} - \dot{m}_{\text{out}} \end{aligned} \tag{2.13}$$

$$\begin{aligned} \{\text{accumulation of energy}\} &= \{\text{energy in}\} - \{\text{energy out}\} \\ \frac{\mathrm{d}E}{\mathrm{d}t} &= \dot{E}_{\text{in}} - \dot{E}_{\text{out}} \end{aligned} \tag{2.14}$$

The constitutive equations are based on theory and heuristics from a variety of fields, such as fluid mechanics, chemical reaction engineering and mineral processing. For example, flow through a constriction is modelled using the Bernoulli relationships from the field of fluid mechanics. Equation 2.15 illustrates such a relationship using F as the flow, k as the constriction resistance, and P as pressure. Naturally, the constitutive equations will be different depending on the particular phenomena being modelled. However, these equations are typically in the form of algebraic equations (AEs).

$$F = k\sqrt{P} \tag{2.15}$$

Lastly, constraints specify, mathematically, the physical limitations of a process. Outside the limits of the constraints, process behaviour is undefined and cannot be modelled or explained by a typical fundamental process model. For example, the percentage that a valve is open may only be between 0% and 100%.

2.2.2.2 Modelling for faults

Given a process model which represents process behaviour under NOC, one may simulate the occurrence of abnormal events by adding fault models to the NOC process model. These fault models typically affect fundamental process model parameters in specific ways depending on the class of fault and the way in which they manifest.

Venkatasubramanian *et al.* (2003*b*) discuss four fault classes where each class refers to the origin of a fault, namely:

1. Gross parameter changes in a model.
2. Structural changes.
3. Malfunctioning sensors.
4. Malfunctioning actuators.

A gross parameter change in a model occurs when a process enters a different operating mode for which the fundamental model parameters are different, either due to set point change or when disturbances not explained by the model occur. A structural change occurs when process equipment (such as pipes and tanks) malfunction, this typically results in a change in correlation structure (and possibly causation) between PVs. Malfunctioning sensors and actuators refer to faults such as sensor bias and sticky valve operation respectively.

Faults may manifest either in an abrupt, incipient, or intermittent manner (Isermann, 2006). This affects how the impact of the fault grows over time with particular emphasis on the basic event causing the fault conditions. An abrupt fault manifests suddenly and typically has a greater impact on process behaviour over a short period of time than an incipient fault, which manifests more slowly. Typically, abrupt faults are more easily detected than incipient faults. For abrupt and incipient faults, the basic event typically persists over time until it is diagnosed and fixed; this is not the case for intermittent faults. An intermittent fault occurs when process behaviour cycles between being normal and abnormal. This causes fault conditions to disappear and appear sporadically, without persisting for over long periods of time. This behaviour makes intermittent faults the most difficult to resolve in comparison to abrupt and incipient faults.

Ideally, a fault diagnosis approach should be evaluated for all possible combinations

of fault class and manifestation with each fault manifesting singly, i.e. only one fault in each set of test data. The use of process simulations enable the user to specify severity, timing, manifestation, and duration of various basic events which allows for the generation of different test data sets under known conditions. Therefore, process simulations can facilitate evaluating the fault diagnosis capability of an approach in a reliable manner.

2.2.3 Evaluation of fault detection performance

Fault detection metrics provide quantitative means to assess fault detection performance given (simulated) process data. The metrics presented here rely on the concepts of false, missing, and true alarms. An **alarm** refers to a fault diagnosis approach reporting that fault conditions are present in a sample of process data. This is referred to as a **false alarm** (false positive) if it occurs under NOC, and as a **true alarm** (true positive) if it occurs under fault conditions. If there is no alarm during fault conditions (when there should be one) then this is referred to as a **missing alarm** (false negative). Table 2.1 illustrates these concepts.

Table 2.1: Illustration of true, false, and missing alarm concepts in fault detection.

Diagnosis \ Actual	Fault	Normal
Alarm	True alarm	False alarm
No alarm	Missing alarm	–

False alarm rate (FAR) is the frequency of a diagnosis result being a false alarm under NOC. It can be calculated, under NOC, by the number of false alarm samples as a fraction of all NOC samples tested from a given data set. Namely:

$$\text{FAR} = \frac{\#\text{samples false alarm}}{\#\text{samples NOC tested}} \tag{2.16}$$

True alarm rate (TAR) is the frequency of a diagnosis result being a true alarm under fault conditions. It can be calculated, under fault conditions, by the number of true alarm samples as a fraction of all fault conditions samples tested from a given data set. Namely:

$$\text{TAR} = \frac{\#\text{samples true alarm}}{\#\text{samples fault conditions tested}} \tag{2.17}$$

The missing alarm rate (MAR) is simply 1 –TAR.

Most fault diagnosis approaches produce some measure which describes the deviation of a sample from NOC. If the deviation from NOC is beyond some threshold for that measure, an alarm is reported. Tuning an approach refers to finding an optimal

threshold such that FAR is minimal and TAR is maximal. In practice, tuning is a difficult task because only NOC data is typically available.[5] Nonetheless, a good fault diagnosis approach should report FAR close to zero and TAR close to one for a wide range of possible threshold values, allowing for some robustness to bad threshold selection.

Irrespective of threshold selection, the fault detection performance for an approach may be summarised by a receiver operating characteristic (ROC) curve. The ROC curve shows the best possible fault detection performance for an approach and gives an indication of the degradation thereof as a result of incorrect threshold selection (Fawcett, 2006; Powers, 2011; Aldrich and Auret, 2013). It is generated by plotting TAR against FAR for various threshold selections. An example of an ROC curve is shown in Figure 2.4.

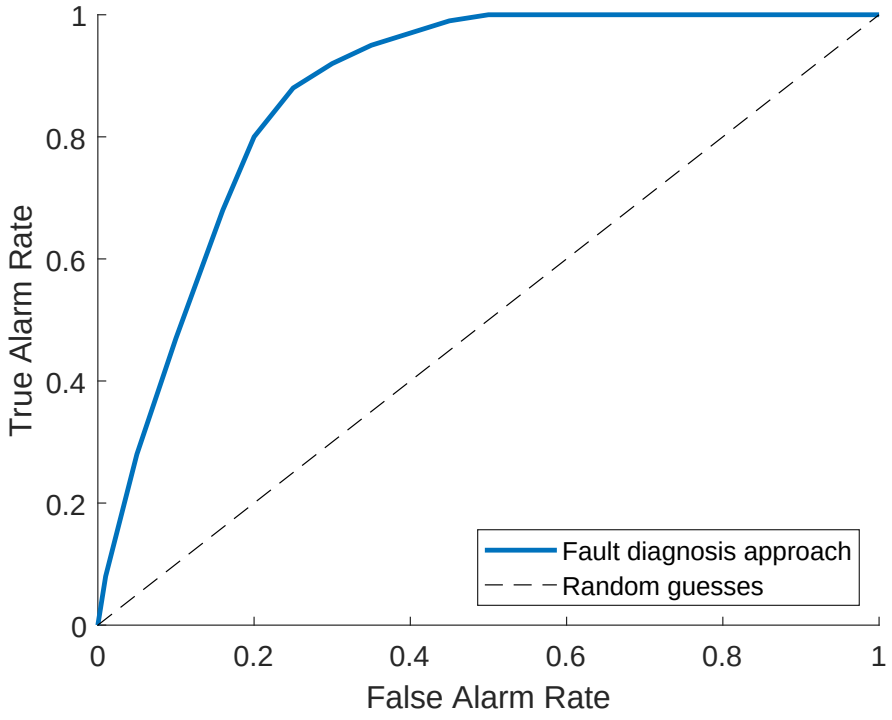


Figure 2.4: Typical receiver operating characteristic (ROC) curve.

The ROC curve is also a good tool for comparison of different fault diagnosis approaches. The shape of the curve is particularly important. If the curve is below or on the diagonal line, then fault detection performance for the approach (regardless of threshold selection) is worse than random chance and the approach is useless for fault detection. Typically, ROC curves are bowed out and away from the diagonal line

[5] Given that simulated process data is used in this thesis, fault data is assumed to be available which allows the research to compare approaches given ideal threshold selection.

as shown in Figure 2.4. For ROC curves of this shape, the closer the curve passes to the (0,1) point (which represents ideal performance), the better the fault detection performance of the approach overall (i.e. regardless of threshold selection).

For ROC curves which are bowed out (to the left) and away from the diagonal line, the area under the ROC curve (AUC) is a useful metric for comparing the fault detection performance of various approaches with one another.[6] The closer the AUC is to 1, the better the fault detection performance of an approach. In contrast, the closer the AUC is to 0.5, the worse the fault detection performance of an approach — an AUC of 0.5 indicates that the ROC curve lies on the diagonal line.

Fawcett (2006) further investigate ROC curves and present a metric which is useful for combining information contained in both FAR and TAR. This metric is called F1 score (Equation 2.18) and it determined from the harmonic mean of precision and TAR, where precision is defined by Equation 2.19.

$$\text{F1 score} = \frac{2 \times \text{precision} \times \text{TAR}}{\text{precision} + \text{TAR}} \tag{2.18}$$

$$\text{precision} = \frac{\#\text{samples true alarm}}{\#\text{samples true alarm} + \#\text{samples false alarm}} \tag{2.19}$$

Maximising F1 score is akin to moving toward the (0,1) point of the ROC curve. A value of one for F1 score indicates that FAR is equal to zero and TAR is equal to one.

2.2.4 Evaluation of root cause analysis performance

The evaluation of root cause analysis is considerably more subjective than that of fault detection. This is because some fault diagnosis approaches, such as MSPM, treat root causes as observed PVs related to a basic event (fault) even though the basic event itself may be a mechanical fault, such as a worn pump impeller. Therefore, for the purposes of this thesis, a root cause need not necessarily be the same as the basic event, but only related closely enough to it.

Note that another challenge is that of fault propagation through a process (Kelly and Lees, 1986). Essentially, a basic event may cause a fault in one area of a process, which in turn causes many fault symptoms to appear throughout the process at different inter-related areas of the process. This makes it difficult to find and assign a particular root cause to a fault since this requires that the fault symptoms be traced back through the inter-related areas of the process; this becomes considerably more difficult if the process contains cyclic structures such as recycle streams or control loops.

[6] Note that AUC is still a useful metric for differently shaped ROC curves, but requires more careful analysis.

Currently we are not aware of any quantitative measures for assessing root cause analysis performance. Therefore, the characteristics in Section 2.2.1 pertaining to root cause analysis are considered instead. Specifically, we consider good isolability and explanation facility most important to the assessment of root cause analysis performance. Good isolability allows a fault diagnosis algorithm to distinguish between different possible root causes and select the most probable one, while good explanation facility describes the propagation of a fault from its root cause to the observed fault symptoms.

Good novelty identifiability and good multiple fault identifiability are also desirable for root cause analysis. A fault diagnosis approach with good novelty identifiability and good root cause analysis performance would be able to distinguish novel basic events from its set of known basic events.[7] A fault diagnosis approach with good multiple fault identifiability and good root cause analysis performance would be able to perform root cause analysis of multiple simultaneous faults. However, in order to have these characteristics, a fault diagnosis approach must produce good root cause analysis performance to begin with. Therefore, good novelty and multiple fault identifiability should have lower priority than good isolability and explanation facility when developing a new approach and comparing different approaches with one another.

2.3 Introduction to Bayesian networks

Model-based PD approaches offer an alternative approach to purely data-driven fault diagnosis. In particular, because the probabilistic models used in these approaches are developed using process knowledge, these approaches should have the potential to provide better root cause analysis performance when compared to a traditional MSPM approach. The PD approach presented in Chapter 3 lies at the core of this work. It also affords the opportunity to familiarise the uninitiated reader with concepts in probability theory using probabilistic techniques in the context of fault diagnosis.

This section begins with a brief introduction to probability theory from a univariate[8] perspective before considering the multivariate case and introducing BNs. This is followed by a discussion of representations for probability distributions over discrete and continuous RVs which are used to demonstrate inference using a simple BN.[9]

[7] An adaptive fault diagnosis approach would also have these novel basic events added into its model to improve root cause analysis in the future.

[8] It is expected that most undergraduate process engineering programmes cover basic probability theory with respect to the univariate case. It is assumed that the reader is familiar with some basic probability theory and if not, the reader is referred to the textbook by Wasserman (2004).

[9] A more comprehensive treatment of BNs may be found in the textbooks by Barber (2011) and Koller and Friedman (2009).

2.3.1 Probability theory and Bayesian networks

Probability distributions are defined over variables or sets of variables. These variables are termed **random variables** to distinguish them from variables in the traditional mathematical sense. The set of possible values that a RV may be assigned is termed its domain. For example, if the RV is the outcome of a coin flip, its domain could be the set $\{heads, tails\}$.

An important distinction can be made between discrete and continuous RVs. Typically, if the number of possible values a RV may assume is finite, it is considered a discrete RV; otherwise it is a continuous[10] RV.

Consider a univariate probability distribution, $\Pr(X)$, over the RV X. It defines a function which maps assignments of X, i.e. $X = x$, to non-negative numbers called probabilities or probability densities. Each $\Pr(X = x)$ is referred to as a **probability** for the discrete case and as a **probability density** for the continuous case. A probability (density), i.e. $\Pr(X = x)$, describes the relative likelihood that X assumes a particular assignment x in its domain. In order to be a legitimate probability distribution, $\Pr(X)$ must satisfy the following criteria:

1. $\Pr(X = x) \geq 0$ for all x in $\text{domain}(X)$
2. $\int_{\text{domain}(X)} \Pr(X = x) = 1$

Note that the second criteria is an integral over the entire domain of X ($\text{domain}(X)$), i.e. all possible values that X may assume. For the discrete case, the integral simplifies to a summation of all the probabilities in the probability distribution, i.e. $\sum_{\text{domain}(X)} \Pr(X = x)$.

The same criteria apply for the multivariate case of probability distributions, where a probability distribution is defined instead over multiple RVs. A multivariate probability distribution defines a function that maps an assignment of values to each RV to a probability (density) in the probability distribution.

As an example, consider the multivariate probability distribution, $\Pr(\mathbf{X})$, over the set of RVs $\mathbf{X} = \{X_1, \ldots, X_k\}$. Each probability (density), $\Pr(\mathbf{X} = \mathbf{x})$, represents the relative likelihood that $\mathbf{X}$ assumes a particular combination of values, $\mathbf{x}$, described as vector of length k containing one assignment, x_i in $\text{domain}(X_i)$, to each X_i, i.e. for $i \in \{1, \ldots, k\}$. This type of probability distribution is also known as a **joint probability distribution (JPD)**.

The probability distribution over one (or more) of the RVs in a JPD, $\Pr(X_j \in \mathbf{X})$, is known as a **marginal probability distribution**. The marginal probability distribution for

[10] It is assumed in this work that the domain of continuous RVs is $\mathcal{R}$.

X_j can be obtained from the JPD, $\Pr(\mathbf{X})$, by **marginalising out** all X_i where $i \neq j$, i.e. by taking the integral of $\Pr(\mathbf{X})$ over the $\text{domain}(X_i)$ for all $i \neq j$:

$$\Pr(X_j) = \int_{\text{domain}(\tilde{\mathbf{X}})} \Pr(\mathbf{X}), \quad \text{where} \quad \tilde{\mathbf{X}} = \{X_i \in \mathbf{X} | i \neq j\}. \tag{2.20}$$

Conditional probability distributions (CPDs) define probability distributions over RVs which are dependent on the values assumed by other RVs. A CPD defines a function that maps a set of values, $\mathbf{y}$, assumed by a set of RVs, $\mathbf{Y}$, to a probability (density) under the condition that the set of RVs, $\mathbf{X}$, which $\mathbf{Y}$ depend upon, assume the set of values $\mathbf{x}$. In other words, for each $\mathbf{X} = \mathbf{x}$ the CPD, $\Pr(\mathbf{Y}|\mathbf{X} = \mathbf{x})$, defines a function describing a probability distribution over $\mathbf{Y}$. Furthermore, the elements of $\mathbf{X}$ are referred to as the **parents** of $\mathbf{Y}$ and the elements of $\mathbf{Y}$ are referred to as the **children** of $\mathbf{X}$.

To better understand the concept of CPDs, consider that one typically expects *warm* weather conditions during *summer* and *cold* weather conditions during *winter*. This relationship is described by the dependency of the RV WEATHER (W) on the RV SEASON (S): in this case S is the parent of W and has the domain $\{\textit{summer}, \textit{winter}\}$ while W has the domain $\{\textit{warm}, \textit{cold}\}$. To reflect the aforementioned intuition, $\Pr(W = \textit{warm}|S = \textit{summer})$ should be higher than $\Pr(W = \textit{cold}|S = \textit{summer})$, and similarly $\Pr(W = \textit{warm}|S = \textit{winter})$ should be lower than $\Pr(W = \textit{cold}|S = \textit{winter})$.

An important relationship exists between JPDs and CPDs which allows CPDs to be used to create JPDs. Let $\mathbf{X}$ and $\mathbf{Y}$ be any two sets of RVs. For an $\mathbf{x}$ such that $\Pr(\mathbf{X} = \mathbf{x}) \neq 0$, the CPD over $\mathbf{Y}$ given $\mathbf{X} = \mathbf{x}$, $\Pr(\mathbf{Y}|\mathbf{X} = \mathbf{x})$, is defined as

$$\Pr(\mathbf{Y}|\mathbf{X} = \mathbf{x}) = \frac{\Pr(\mathbf{Y}, \mathbf{X} = \mathbf{x})}{\Pr(\mathbf{X} = \mathbf{x})}. \tag{2.21}$$

The JPD over $\mathbf{X}$ and $\mathbf{Y}$ may be obtained by rearranging Equation 2.21 and multiplying $\Pr(\mathbf{Y}|\mathbf{X} = \mathbf{x})$ by $\Pr(\mathbf{X} = \mathbf{x})$ for all $\mathbf{X} = \mathbf{x}$. One can then specify those CPDs instead of working directly with JPDs; the former is more convenient than the latter.

Equation 2.21 also enables the decomposition of a JPD into a product of CPDs which is a vital aspect of BNs and is known as the **chain rule of probability**. Consider the JPD, $\Pr(\mathbf{X})$, over the set of RVs $\mathbf{X} = \{X_1, \ldots, X_k\}$. The JPD may be arbitrarily

decomposed into a product of CPDs by repeatedly applying Equation 2.21:

$$\begin{aligned}\Pr(\mathbf{X}) &= \Pr(X_1, \ldots, X_k) \\ &= \Pr(X_1) \Pr(X_2, \ldots, X_k | X_1) \\ &= \Pr(X_1) \Pr(X_2|X_1) \Pr(X_3, \ldots, X_k | X_1, X_2) \\ &= \Pr(X_1) \Pr(X_2|X_1) \ldots \Pr(X_{k-1}|X_1, \ldots, X_{k-2}) \Pr(X_k|X_1, \ldots, X_{k-1})\end{aligned} \tag{2.22}$$

This decomposition is arbitrary because in practice, some of the RVs in $\mathbf{X}$ may be **independent** of one another and would have no influence on one another. To illustrate this concept of independence, let $\mathbf{X}_a$ and $\mathbf{X}_b$ be two disjoint subsets of $\mathbf{X}$. $\mathbf{X}_a$ and $\mathbf{X}_b$ are independent of one another, if

$$\Pr(\mathbf{X}_a, \mathbf{X}_b) = \Pr(\mathbf{X}_a) \Pr(\mathbf{X}_b). \tag{2.23}$$

This implies that

$$\Pr(\mathbf{X}_a | \mathbf{X}_b = \mathbf{x}_b) = \frac{\Pr(\mathbf{X}_b = \mathbf{x}_b, \mathbf{X}_a)}{\Pr(\mathbf{X}_b = \mathbf{x}_b)} = \frac{\Pr(\mathbf{X}_b = \mathbf{x}_b) \Pr(\mathbf{X}_a)}{\Pr(\mathbf{X}_b = \mathbf{x}_b)} = \Pr(\mathbf{X}_a), \tag{2.24}$$

and

$$\Pr(\mathbf{X}_b | \mathbf{X}_a = \mathbf{x}_a) = \frac{\Pr(\mathbf{X}_a = \mathbf{x}_a, \mathbf{X}_b)}{\Pr(\mathbf{X}_a = \mathbf{x}_a)} = \frac{\Pr(\mathbf{X}_a = \mathbf{x}_a) \Pr(\mathbf{X}_b)}{\Pr(\mathbf{X}_a = \mathbf{x}_a)} = \Pr(\mathbf{X}_b). \tag{2.25}$$

Accounting for the independences between RVs, the chain rule in Equation 2.22 can be summarised by

$$\Pr(\mathbf{X}) = \Pr(X_1, \ldots, X_k) = \prod_{i=1}^{k} \Pr(X_i | \mathrm{par}(X_i)) \tag{2.26}$$

where $\mathrm{par}(X_i)$ are the parents of X_i.

CPDs can be used to describe the relationship between multiple parent RVs ($\mathbf{X}$) and one child RV (Y) much like in an AE where $y = f(\mathbf{x})$. This makes CPDs useful for encoding individual process model equations which, when put together using the chain rule, describe the full process model as a JPD.

As an example, consider the measurement of liquid level in a tank as shown in Figure 2.5. Two RVs L and L_m represent the actual liquid level and the sensor reading, respectively.

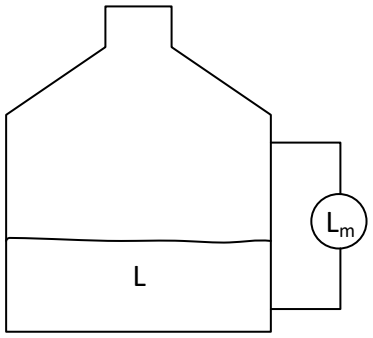


Figure 2.5: Schematic for liquid level example. L and L_m represent the actual liquid level and the sensor reading respectively.

A third RV S_L represents the status of the sensor which may be *normal* or *faulty*. The actual liquid level L is independent of the sensor status S_L and the sensor reading L_m is dependent on both L and S_L. Therefore the JPD, $\Pr(L, L_m, S_L)$ may be decomposed as

$$\begin{aligned}\Pr(L, L_m, S_L) &= \Pr\left(L|L_m, S_L\right) \Pr\left(S_L|L, L_m\right) \Pr\left(L_m|L, S_L\right) \\ &= \Pr(L) \Pr(S_L) \Pr\left(L_m|L, S_L\right)\end{aligned} \tag{2.27}$$

by taking advantage of the independence structure between the RVs.

This decomposition may be represented graphically by drawing a node for each RV and drawing an edge between dependent RVs. The direction of each edge is from the RVs on the right of the conditional bar (|) to the RVs on its left, i.e. from parent to child. Applying this procedure to the liquid level example leads to Figure 2.6, where discrete and continuous RVs are represented by rectangles and circles respectively. The combination of Figure 2.6 and the CPDs for each RV is called a **Bayesian network** (Pearl, 1985).

Definition 2.1: Bayesian network (BN)

Let $\mathbf{X}$ be a set of RVs $\{X_1, \ldots, X_k\}$. Let $\mathcal{G}$ be a directed acyclic graph (Thulasiraman and Swamy, 1992) defined by the tuple, $\langle X, E\rangle$, of nodes X and edges E such that one node is associated with each RV in $\mathbf{X}$. Let $\boldsymbol{\theta}$ be a set of CPDs $\{\theta_1, \ldots, \theta_k\}$, one associated with each RV $X_i \in \mathbf{X}$, such that $\theta_i \in \boldsymbol{\theta}$ is the CPD: $\Pr\left(X_i|\text{par}(X_i)\right)$ whose dependence structure is defined by $\mathcal{G}$.

A BN, $\mathcal{B}$, is a compact representation of the JPD, $\Pr(\mathbf{X})$, defined by the tuple: $\langle \mathcal{G}, \boldsymbol{\theta}\rangle$. The JPD may be computed using Equation 2.26.

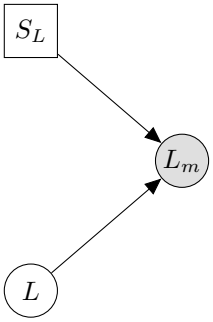


Figure 2.6: BN for liquid level example. L and L_m represent the actual liquid level and the sensor reading respectively and S_L is the status of the sensor: *normal* or *faulty*. Shading on L_m indicates that it is an **observed** RV (i.e. we will condition on its value based on a sensor reading).

Typically, CPDs are defined in the causal direction such that a direct causal relationship exists from the set of RVs on the right of the conditional bar to that on its left. **Bayes' theorem** is a logical corollary of Equation 2.21 which enables one to compute a CPD with reversed dependency, i.e. $\Pr\left(\mathbf{X}|\mathbf{Y}\right)$ instead of $\Pr\left(\mathbf{Y}|\mathbf{X}\right)$. Since we have

$$\Pr\left(\mathbf{X}|\mathbf{Y}\right)\Pr\left(\mathbf{Y}\right) = \Pr\left(\mathbf{X},\mathbf{Y}\right) = \Pr\left(\mathbf{Y}|\mathbf{X}\right)\Pr\left(\mathbf{X}\right), \tag{2.28}$$

Bayes theorem follows:

$$\Pr\left(\mathbf{X}|\mathbf{Y}\right) = \frac{\Pr\left(\mathbf{Y}|\mathbf{X}\right)\Pr\left(\mathbf{X}\right)}{\Pr\left(\mathbf{Y}\right)}. \tag{2.29}$$

When $\mathbf{X}$ is a set of causes and $\mathbf{Y}$ is a set of effects, then Bayes' theorem enables **inference** – reasoning from effects to causes in the presence of uncertainty.

Consider again the liquid level example of Figure 2.5 whose BN defines relationships between the sensor status (cause) and the sensor reading (effect). Assume that a normally functioning sensor produces a noisy reading of the actual liquid level, while a faulty sensor produces an arbitrary reading irrespective of the actual liquid level. Given that a sensor reading is observed, Bayes' theorem may be used to reason whether or not that particular reading was the result of a normally functioning sensor or a faulty one. This may be phrased as the query: "What is the probability distribution over the sensor status given observation of the sensor reading?" and can be formulated using Bayes' theorem as

$$\Pr\left(S_L|L_m = l_m\right) = \frac{\Pr\left(L_m = l_m|S_L\right)\Pr\left(S_L\right)}{\Pr\left(L_m = l_m\right)}. \tag{2.30}$$

The numerator on the right-hand side is the product of the likelihood[11] function

[11]Note that we use unnormalised likelihood as a surrogate for the conditional probability of observation.

$\Pr\left(L_m = l_m | S_L\right)$ and the prior $\Pr\left(S_L\right)$. The **likelihood function** specifies the dependence of the sensor reading on the sensor status, i.e. the probability that any particular sensor reading is observed given that the sensor status is known. The **prior** is the probability distribution over the sensor status before the sensor reading is observed. The denominator is the **normalisation constant** $\Pr\left(L_m = l_m\right)$. It is the marginal probability of observing any particular sensor value. It normalises the numerator such that the posterior on the left-hand side, $\Pr\left(S_L | L_m = l_m\right)$, sums to 1. The **posterior** is the probability distribution which answers the query. Using these terms, Bayes' theorem may be rewritten as

$$\text{posterior} = \frac{\text{likelihood function} \times \text{prior}}{\text{normalisation constant}}. \tag{2.31}$$

All of the probability distributions in Equation 2.30 can be computed from the JPD, $\Pr\left(L, L_m, S_L\right)$. The numerator may be computed by marginalising out L from the JPD and the denominator may be computed by marginalising out L and S_L from the JPD. Recalling that the JPD may be computed as a product of CPDs, the query in Equation 2.30 becomes:

$$\Pr\left(S_L | L_m = l_m\right) = \frac{\int_L \Pr\left(L_m = l_m | L, S_L\right) \Pr\left(L\right) \Pr\left(S_L\right)}{\int_{L,S_L} \Pr\left(L_m = l_m | L, S_L\right) \Pr\left(L\right) \Pr\left(S_L\right)}. \tag{2.32}$$

Section 2.3.2 presents some typical representations of probability distributions which are used for a numerical example in Section 2.3.3 demonstrating the mechanism for computing Equation 2.32.

2.3.2 Representation of probability distributions

This subsection considers the representation of different types of probability distributions. Note that explanations here are typically illustrative of use-cases in this work and do not always apply to the general case.

Discrete case Probability distributions over discrete RVs are typically represented as tables. Recall that for a discrete RV, A, the number of values, a_i in $\text{domain}\left(A\right)$, it may assume form a finite or countable set. The table describing the probability distribution of A contains the probability $\Pr\left(A = a_i\right)$ for a single value a_i to be assumed by A.

In the multivariate case, a probability distribution table contains a probability for each assignment of values to the discrete RVs. Consider the JPD, $\Pr\left(A, B, C\right)$, over the set of binary discrete RVs $\{A, B, C\}$. There exists one entry, $\Pr\left(A = a_i, B = b_i, C = c_i\right)$, in a table for each combination of assignments, a_i, b_i, and c_i, to A, B, and C, respectively. An example probability distribution containing these three RVs is shown in Table 2.2.

Table 2.2: An example probability distribution over a set of 3 binary RVs.

A	B	C	$\Pr(A, B, C)$
a_1	b_1	c_1	0.16
a_1	b_1	c_2	0.20
a_1	b_2	c_1	0.06
a_1	b_2	c_2	0.10
a_2	b_1	c_1	0.10
a_2	b_1	c_2	0.16
a_2	b_2	c_1	0.18
a_2	b_2	c_2	0.04

As another example, the marginal probability distribution over A, $\Pr(A)$, may be obtained from the JPD by summing the entries in the JPD for which $A = a_i$ for each a_i in $\text{domain}(A)$. Using the probability distribution in Table 2.2, the first entry in the marginal probability distribution of A is

$$\Pr(A = a_1) = 0.16 + 0.2 + 0.06 + 0.1 = 0.52. \tag{2.33}$$

This illustrates procedure described in Equation 2.20 and holds for obtaining the marginal probability distributions over multiple discrete RVs as well.

Continuous case Probability distributions over continuous RVs are often represented as functions. This is because the number of values that a continuous RV X may assume is infinite. It is assumed in this work that these values are all real values. One particularly common function used to represent $\Pr(X = x)$ is the Gaussian function.[12] The Gaussian function is parametrised by the mean, μ, and variance, σ^2, of X and is written as

$$\Pr(X = x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right). \tag{2.34}$$

Note that $\Pr(X = x)$ is actually a probability density. Probabilities for continuous RVs can only be computed for sets of values – this is done by integrating the probability density function (PDF) over the set of values.

A Gaussian distributed continuous RV has its probability distribution completely defined by its mean, μ, and variance, σ^2. Therefore, the shorthand: $\mathcal{N}(X : \mu, \sigma^2)$ is used to refer to such a RV, X, with mean μ and variance σ^2. A similar shorthand holds for the multivariate case as well, but instead μ is a vector of means ($\boldsymbol{\mu}$) and σ^2 is a square covariance matrix[13] ($\boldsymbol{\Sigma}$).

[12] Note that we will typically only use the Gaussian function to represent continuous probability distributions in this work.

[13] Note that $\boldsymbol{\Sigma}$ is always a symmetric positive semi-definite matrix.

To illustrate, let $\mathbf{X}$ be a set of continuous RVs $\{X_1, \ldots, X_k\}$ with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. $\boldsymbol{\mu}$ is a vector of length k where each entry μ_i in $\boldsymbol{\mu}$ is the mean of the RV $X_i \in \mathbf{X}$. $\boldsymbol{\Sigma}$ is a square matrix of size $k \times k$ where each entry along its diagonal, i.e. each σ_i^2 in $\operatorname{diag}(\boldsymbol{\Sigma})$, is the variance of the RV X_i and each of the off-diagonal entries, $\sigma_{i,j}^2$ where $i \neq j$, in $\boldsymbol{\Sigma}$ is the covariance between the RVs X_i and X_j. The probability density of the vector of assignments $\mathbf{x}$ to $\mathbf{X}$ is

$$\Pr(\mathbf{X} = \mathbf{x}) = \frac{1}{\sqrt{\det(2\pi\boldsymbol{\Sigma})}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})\right). \tag{2.35}$$

The shorthand $\mathcal{N}(\mathbf{X} : \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is used to refer to a set of Gaussian distributed continuous RVs, $\mathbf{X}$, with mean vector $\boldsymbol{\mu}$, and covariance matrix $\boldsymbol{\Sigma}$.

Hybrid case Hybrid probability distributions are defined over a mixture of discrete and continuous RVs. For each combination of values assumed by the set of discrete RVs, a hybrid probability distribution typically contains one probability and one conditional PDF over the set of continuous RVs. If all those PDFs are Gaussian functions, then this probability distribution closely resembles a **mixture of Gaussians** (Koller and Friedman, 2009), however, since there are discrete RVs involved we will refer to such a probability distribution as a hybrid mixture of Gaussians. In the context of this hybrid mixture of Gaussians, the relative likelihoods are referred to as **weights** and the functions are referred to as **mixture components**.

Consider the hybrid mixture of Gaussians, $\Pr(A, B, \mathbf{X})$, defined over the set of binary discrete RVs $\{A, B\}$ and the set of continuous RVs $\mathbf{X} = \{X_1, \ldots, X_k\}$. This hybrid probability distribution may be represented as a table containing one Gaussian mixture component $\Pr(\mathbf{X}|A = a_i, B = b_i)$ and one probability $\Pr(A = a_i, B = b_i)$ for each combination of assignments, a_i and b_i, to A and B as shown in Table 2.3.

Table 2.3: An example probability distribution for a hybrid mixture of Gaussians.

A	B	$\Pr(A, B)$	$\Pr(\mathbf{X}\|A, B)$
a_1	b_1	0.10	$\mathcal{N}(\mathbf{X} : \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$
a_1	b_2	0.20	$\mathcal{N}(\mathbf{X} : \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$
a_2	b_1	0.25	$\mathcal{N}(\mathbf{X} : \boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$
a_2	b_2	0.45	$\mathcal{N}(\mathbf{X} : \boldsymbol{\mu}_4, \boldsymbol{\Sigma}_4)$

Continuous RVs can be marginalised out from the hybrid mixture of Gaussians distribution applying the marginalisation procedure to each of the mixture components. Marginalising discrete RVs out from the hybrid mixture of Gaussians is more complicated since the number of mixture components will be reduced as a result thereof. This process is known as **collapsing** a hybrid mixture of Gaussians into

one containing fewer mixture components and can be done approximately using the M-projection method (Lauritzen, 1996; Koller and Friedman, 2009).

The M-projection method approximates the collapsed hybrid mixture of Gaussians by a Gaussian which maintains the same overall mean and overall covariance matrix as the original hybrid mixture of Gaussians. It does this by computing a weighted average mean vector and covariance matrix of the mixture components which need to be collapsed into a Gaussian. If only a subset of the discrete RVs need to be marginalised out from the hybrid mixture of Gaussians, then the procedure for marginalising out discrete RVs from a discrete JPD determines which mixture components need to be collapsed together – recall that the hybrid mixture of Gaussians contains one Gaussian for each combination of assignments to the discrete RVs. Also note that when marginalising out a subset discrete RVs, the probabilities associated with the mixture components being collapsed cannot be directly used as weights during M-projection and need to be normalised first, i.e. to preserve the weighted average aspect of M-projection.

To collapse a set of mixture components, let w_i be the weight of the i-th Gaussian, $\mathcal{N}(\mathbf{X} : \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, in a set of n Gaussians. The mixture components may be collapsed into a single Gaussian, $\mathcal{N}\left(\mathbf{X} : \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}}\right)$, according to Equations 2.36 and 2.37.

$$\tilde{\boldsymbol{\mu}} = \sum_{i=1}^{n} w_i \boldsymbol{\mu}_i \tag{2.36}$$

$$\tilde{\boldsymbol{\Sigma}} = \sum_{i=1}^{n} w_i \boldsymbol{\Sigma}_i + \sum_{i=1}^{n} w_i \left(\boldsymbol{\mu}_i - \tilde{\boldsymbol{\mu}}\right)\left(\boldsymbol{\mu}_i - \tilde{\boldsymbol{\mu}}\right)^{\mathrm{T}} \tag{2.37}$$

Conditional cases CPDs define a probability distribution over a set of RVs for each combination of values assumed by the set of RVs being conditioned on.

Discrete CPDs Consider the CPD, $\Pr\left(A, B|C\right)$, defined over the set of binary discrete RVs $\{A, B, C\}$. A probability distribution over A and B can be defined for each assignment c_i to C as shown in Table 2.4.

Table 2.4: An example CPD over a set of binary RVs. Note that $\Pr\left(A, B|C = c_i\right)$ sum to one for each c_i.

C	A	B	$\Pr\left(A, B\|C\right)$
c_1	a_1	b_1	0.32
c_1	a_1	b_2	0.12
c_1	a_2	b_1	0.20
c_1	a_2	b_2	0.36
c_2	a_1	b_1	0.40
c_2	a_1	b_2	0.20
c_2	a_2	b_1	0.32
c_2	a_2	b_2	0.08

The JPD over $\{A, B, C\}$ is the product of $\Pr\left(A, B|C\right)$ and $\Pr\left(C\right)$ according to Equation 2.26. The probability distribution $\Pr\left(C\right)$ weights the probability distributions $\Pr\left(A, B|C = c_1\right)$ and $\Pr\left(A, B|C = c_2\right)$. If these distributions are equally weighted, i.e. $\Pr\left(C = c_i\right) = 0.5$ for c_1 and c_2, then the JPD, $\Pr\left(A, B, C\right)$, is the same as that in Table 2.2 and may be obtained by multiplying each entry in Table 2.4 by 0.5 and summing entries which agree on their assignments to A and B.

Discrete CPDs will be useful for representing the probability distributions of process component statuses over time. For example the probability of a sensor transitioning from a *normal* status to a *faulty*, between consecutive observations, should be relatively low whereas the probability of the sensor keeping the *normal* status should be relatively high.

Continuous CPDs Although alternative forms of purely continuous CPDs exist, a commonly used (since it is computationally convenient) representation of continuous CPDs is the linear Gaussian (Koller and Friedman, 2009).

The linear Gaussian representation assumes a linear relationship between the mean(s) of one (multivariate) Gaussian $\mathcal{N}\left(\mathbf{X} : \boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$ and another univariate[14] Gaussian $\mathcal{N}\left(Y : \mu_y, \sigma_y^2\right)$. We assume that Y and $\mathbf{X}$ are related by the formula

$$Y = \beta_0 + \boldsymbol{\beta}^{\mathrm{T}}\mathbf{X} + \epsilon, \tag{2.38}$$

where $\epsilon : \mathcal{N}\left(0, \sigma_0^2\right)$. It then follows that

$$\Pr\left(Y|\mathbf{X} = \mathbf{x}\right) : \mathcal{N}\left(Y : \beta_0 + \boldsymbol{\beta}\mathbf{x}, \sigma_0^2\right) \tag{2.39}$$

and

$$\Pr\left(Y\right) : \mathcal{N}\left(Y : \mu_y = \beta_0 + \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{\mu}, \sigma_y^2 = \sigma_0^2 + \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{\Sigma}\boldsymbol{\beta}\right). \tag{2.40}$$

[14]Note that it is assumed that this is always univariate in this thesis. This assumption is justified by the use of BN representations for JPDs – we use Equation 2.26 and a just-in-time linearisation method described in Section 3.4.2 to do this.

Note that $\boldsymbol{\beta}$ is a vector of coefficients, containing one β_i for each RV $X_i \in \mathbf{X}$.

The JPD over $\mathbf{X}$ and Y, $\Pr(\mathbf{X}, Y)$, requires computation of the covariance between each X_i and Y, $\operatorname{cov}(X_i, Y)$. The mean vector of the JPD is simply a concatenation of $\boldsymbol{\mu}$ and μ_y and its covariance matrix is a square matrix containing $\boldsymbol{\Sigma}$ and σ_y^2 as well as some additional covariance terms, i.e.

$$\Pr(\mathbf{X}, Y) : \mathcal{N}\left(Y : \begin{bmatrix}\boldsymbol{\mu} \\ \mu_y\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma} & \vdots \\ \cdots & \sigma_y^2\end{bmatrix}\right). \tag{2.41}$$

The remaining entries of the covariance matrix, $\operatorname{cov}(X_i, Y)$, are calculated as:

$$\operatorname{cov}(X_i, Y) = \sum_j \beta_j \boldsymbol{\Sigma}_{i,j}. \tag{2.42}$$

Continuous CPDs are useful for representing relationships between continuous RVs, such as the dependence between a PV and a noisy sensor reading.

Given a multivariate Gaussian over two sets of RVs $\mathbf{Q}$ and $\mathbf{O}$ representing the PVs and sensor readings respectively, one may also condition the Gaussian to determine $\Pr(\mathbf{Q}|\mathbf{O} = \mathbf{o})$. In particular, if $\Pr(\mathbf{Q}, \mathbf{O})$ is represented by

$$\mathcal{N}\left(\mathbf{Q}, \mathbf{O} : \begin{bmatrix}\boldsymbol{\mu}_\mathbf{Q} \\ \boldsymbol{\mu}_\mathbf{O}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_{\mathbf{QQ}} & \boldsymbol{\Sigma}_{\mathbf{QO}} \\ \boldsymbol{\Sigma}_{\mathbf{OQ}} & \boldsymbol{\Sigma}_{\mathbf{OO}}\end{bmatrix}\right), \tag{2.43}$$

then $\Pr(\mathbf{Q}|\mathbf{O} = \mathbf{o})$ is also a Gaussian, represented by $\mathcal{N}\left(\mathbf{Q}|\mathbf{O} = \mathbf{o} : \boldsymbol{\mu}'_\mathbf{Q}, \boldsymbol{\Sigma}'_{\mathbf{QQ}}\right)$. Here $\boldsymbol{\mu}'_\mathbf{Q}$ and $\boldsymbol{\Sigma}'_{\mathbf{QQ}}$ are determined by Equations 2.44 and 2.45 respectively (Eaton, 1983).

$$\boldsymbol{\mu}'_\mathbf{Q} = \boldsymbol{\mu}_\mathbf{Q} + \boldsymbol{\Sigma}_{\mathbf{QO}}\boldsymbol{\Sigma}_{\mathbf{OO}}^{-1}(\mathbf{o} - \boldsymbol{\mu}_\mathbf{O}) \tag{2.44}$$

$$\boldsymbol{\Sigma}'_{\mathbf{QQ}} = \boldsymbol{\Sigma}_{\mathbf{QQ}} - \boldsymbol{\Sigma}_{\mathbf{QO}}\boldsymbol{\Sigma}_{\mathbf{OO}}^{-1}\boldsymbol{\Sigma}_{\mathbf{OQ}} \tag{2.45}$$

Hybrid CPDs Consider again the hybrid mixture of Gaussians representation of Table 2.3 on page 35. Each Gaussian in the mixture is a hybrid CPD, a probability distribution over a set of continuous RVs $\mathbf{X}$ given a combination of assignments to a set of discrete RVs $\{A, B\}$. Each particular combination of values assumed by the set of discrete RVs selects one of the Gaussians in the mixture.

Such a hybrid CPD is useful for representing relationships between the status of process components, such as sensors and actuators, and PVs, such as pressures and flows. The status of the components selects the operational mode of a process, defined by relationships between PVs, which affects the likelihood of observing a particular set of sensor readings.

It should be noted that the case of augmented CPDs, where discrete RVs depend on continuous RVs, is considerably more complex and since augmented CPDs are not used in this research, they are not further discussed here.

2.3.3 Liquid level inference example (continued)

Consider again the liquid level example of Figure 2.5 on page 30. Recall that the goal was to compute the query (probability) described by Equation 2.32. The task of computing this query was referred to as inference because probable causes were being inferred from observed effects. Namely, in this example, the sensor reading, L_m, is an observed RV and the status of the sensor, S_L, is unknown.

The probability distributions in Table 2.5 are assumed for purpose of the example. Note that assumptions include that: the tank is half-full; the actual liquid level in the tank does not change; the level sensor takes independent measurements of the liquid level in the tank; and that the status of the level sensor is initially unknown.

Table 2.5: Liquid level example probability distributions in BN. L and L_m represent the actual liquid level and the sensor reading, as a percentage full, respectively. S_L is the status of the sensor: *normal* or *faulty*.

S_L	$\Pr(S_L)$	$\Pr(L_m \mid S_L, L)$
normal	0.5	$\mathcal{N}\left(\mu_{L_m} = \mu_L, \sigma^2_{L_m} = 6.25 + \sigma^2_L\right)$
faulty	0.5	0.01 for $0 \leq l_m \leq 100$

$$\Pr(L) = \mathcal{N}\left(\mu_L = 50, \sigma^2_L = 0.25\right)$$

$\Pr\left(L_m \mid S_L = \textit{normal}\right)$ is obtained by substituting the mean and variance of $\Pr(L)$ into $\Pr\left(L_m \mid S_L = \textit{normal}, L\right)$ as in Equation 2.46 – a special case of Equation 2.40 with $\boldsymbol{\beta} = [1]$.

$$\mathcal{N}\left(L_m : \mu_{L_m} = 50, \sigma^2_{L_m} = 6.5\right) \tag{2.46}$$

Figure 2.7 is a visualisation of $\Pr\left(L_m \mid S_L\right)$ for the cases of $S_L = \textit{normal}$ and $S_L = \textit{faulty}$ when $L = 50\%$.

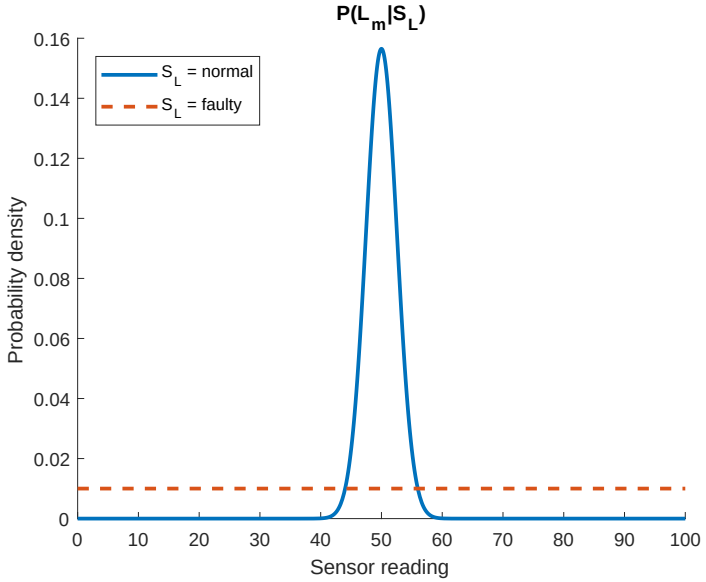


Figure 2.7: Probability distributions of sensor reading given different sensor statuses. The solid line and dashed lines represent $\Pr\left(L_m|S_L = \textit{normal}\right)$ and $\Pr\left(L_m|S_L = \textit{faulty}\right)$ respectively.

Suppose a sensor reading is taken, and the sensor reads a value of 52% as indicated in Figure 2.8.

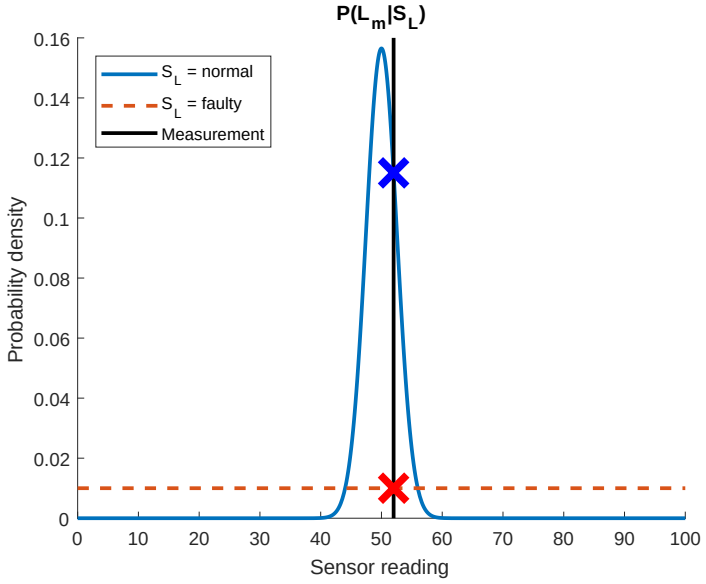


Figure 2.8: Probability distributions of sensor reading given different sensor statuses and an observation of 52% sensor reading.

Before the reading, the status of the sensor is unknown and there is a uniformly distributed **belief** (i.e. a probability distribution) over its two possible values. The goal is to update this belief in accordance with the observation, but this process is not immediately clear without the mechanism of Bayes' theorem. Using the current belief, $\Pr(S_L)$, as the prior in Equation 2.30, the posterior belief, $\Pr(S_L|L_m = 52\%)$, may be computed as follows.

For each case of sensor status, the product of the likelihood function and prior, i.e. the numerator of Equation 2.30, needs to be computed.

Consider the case when the sensor status is *normal*. Applying Equation 2.34 to $\Pr(L_m|S_L = \textit{normal})$, the probability density for L_m assuming the value 52% is 0.1150. Thus, the value of the numerator of Equation 2.30 for this case is:

$$\Pr(L_m = 52\%|S_L = \textit{normal}) \Pr(S_L = \textit{normal}) = 0.1150 \times 0.5 = 0.0575. \quad (2.47)$$

Next, consider the case when the sensor status is *faulty*. The probability density for L_m assuming the value 52% is 0.01 due its uniform probability distribution. The value of the numerator of Equation 2.30 for this case is:

$$\Pr(L_m = 52\%|S_L = \textit{faulty}) \Pr(S_L = \textit{faulty}) = 0.01 \times 0.5 = 0.005. \quad (2.48)$$

The normalization constant, $\Pr(L_m = 52\%)$, is the sum of the above two values, i.e. Equations 2.47 and 2.48.

$$\Pr(L_m = 52\%) = 0.0575 + 0.005 = 0.0625 \quad (2.49)$$

Dividing Equations 2.47 and 2.48 by Equation 2.49 yields the posterior probability distribution of the sensor status as shown in Table 2.6.

Table 2.6: Posterior belief of sensor status after observation for a sensor reading of 52%.

S_L	$\Pr(S_L\|L_m = 52\%)$
normal	0.92
faulty	0.08

Table 2.6 shows that the probability (or belief) that the sensor is functioning *normally* has been strengthened by the observation. This is primarily due to the probability density of $\Pr(L_m = 52\%|S_L = \textit{normal})$ being much greater than that of $\Pr(L_m = 52\%|S_L = \textit{faulty})$, i.e. $0.1150 \gg 0.01$.

Now suppose that another reading is taken (independent of the previous reading), and the sensor reads a value of 57% as shown in Figure 2.9. How does the new sensor reading affect the previously updated belief in Table 2.6?

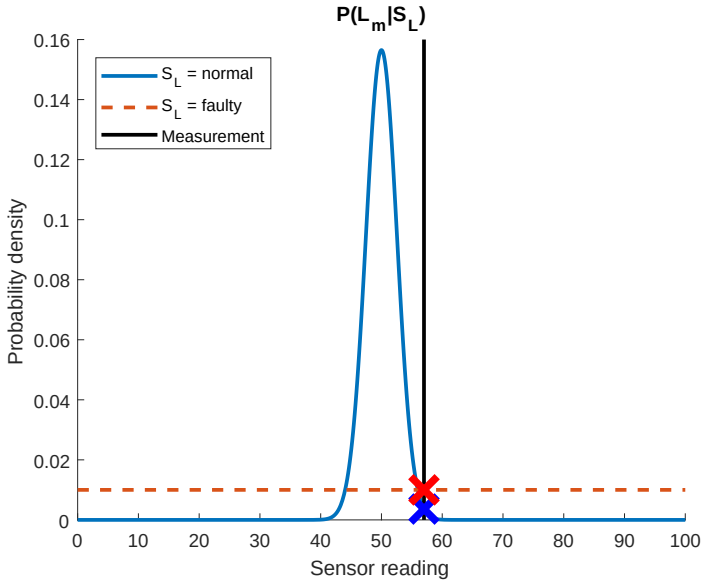


Figure 2.9: Probability distributions of sensor reading given different sensor statuses and observation of 57% sensor reading.

Starting with the current belief, in Table 2.6, as the prior in Equation 2.30, calculations similar to those in Equations 2.47, 2.48, and 2.49 ensue to compute the new posterior, $\Pr\left(S_L|L_m = 57\%\right)$. The calculations are performed as follows:

$$\Pr\left(L_m = 57\%|S_L = \textit{normal}\right)\Pr\left(S_L = \textit{normal}\right) = 0.0036 \times 0.92 = 0.0033 \tag{2.50}$$

$$\Pr\left(L_m = 57\%|S_L = \textit{faulty}\right)\Pr\left(S_L = \textit{faulty}\right) = 0.01 \times 0.08 = 0.0008 \tag{2.51}$$

$$\Pr\left(L_m = 57\%\right) = 0.0033 + 0.0008 = 0.0041 \tag{2.52}$$

Table 2.7: Updated posterior belief of sensor status after observation for a sensor reading of 57%.

S_L	$\Pr\left(S_L\|L_m = 57\%\right)$
normal	0.806
faulty	0.194

The result of updating the previous belief with the new observation (Table 2.7) now shows a continued belief that the sensor status is *normal*. This is because, even

though the probability density of $\Pr\left(L_m = 57\%|S_L = \textit{faulty}\right)$ is greater than that of $\Pr\left(L_m = 57\%|S_L = \textit{normal}\right)$, the belief that the sensor is faulty is considered far less likely by the prior belief. Since it was strongly believed that the sensor status is *normal*, a new observation would have to be further away from the mean of the probability distribution $\Pr\left(L_m|S_L = \textit{normal}\right)$ in order for the belief to report a relatively higher probability for the sensor being *faulty*.

In contrast, if the prior belief for the second sensor reading had not been updated to reflect the posterior belief after the first sensor reading, i.e. if the initial belief in Table 2.5 was maintained, then the posterior belief in Table 2.7 would have reported that sensor is *faulty* with probability 0.735.

It can be concluded that previous observations influence how a belief is updated with new observations in the future. This insight plays a large role in the diagnosis of fault conditions in processes where sensor readings are continuously monitored. In this case, the sensors provide a stream of observations which can be used to maintain and update beliefs about the operating mode of the process as well as the distributions of the underlying PVs. This idea is further discussed in Chapter 3 by presentation of a PD approach based on DBNs.

Chapter 3

Probabilistic Fault Diagnosis Approach

Chapter 2 presented some background theory including a traditional MSPM fault diagnosis approach, tools and methods for benchmarking fault diagnosis capability of an approach, and some introductory concepts from probability theory and BNs. This chapter presents a mixture of background theory and methodology for the development and implementation of a DBN-based PD approach based on Lerner *et al.* (2000) and Lerner (2002).

The chapter begins with a formalisation of DBNs in the context of this thesis, including relevant assumptions. Next, the problem of exponential blow-up in the sizes of JPDs with respect to time in a DBN is discussed and an approach, i.e. an inference engine, tackling this problem is presented. The use of this inference engine for fault diagnosis is also detailed. This is followed by several enhancements which attempt to reduce the computational requirements of the inference engine and concludes the detailed presentation of concepts involved in the PD approach. Finally, methods used for the development of DBNs for process monitoring purposes are detailed before the chapter is concluded. This chapter also provides background for the discussion of other DBN-based PD approaches presented in literature in the next chapter

3.1 Dynamic Bayesian networks

Section 2.3 introduced some basic probability theory concepts as well as BNs, and then demonstrated how probabilistic reasoning may be used to detect possible fault conditions in a simple example involving the measurement of liquid level in a closed tank. Conceptually, this demonstration can extend to processes which have many more process components (such as valves, sensors, pumps, etc.), each with their own discrete statuses, and many more PVs (such as pressures, flows, temperatures, etc.). However, the statuses of process components and values of PVs typically change over time. To account for this, discrete-time stochastic processes may be represented by DBNs — a tool for modelling RVs changing over time.

This section presents some background theory on DBNs (including assumptions relevant to this thesis) before introducing an example case study, based on the one in Section 2.3, which will be later used to showcase the functionality of the PD approach.[1]

3.1.1 Theoretical background

For a particular point in time, t, all of the RVs that represent the statuses of process components and values of PVs are referred to as the **state** of a process. In a DBN, the state at time t is denoted as $\mathbf{X}^{(t)}$ where each $X_i^{(t)} \in \mathbf{X}^{(t)}$ is one RV in the state.

Collectively these RVs are referred to as being in the **time slice** t because, in this setting, t is typically restricted to only take integer values for which the difference in real time between any two consecutive time slices, t and $t+1$, is constant. Furthermore, time is only allowed to move forward from an initial process state at time slice 0 – this is represented by the probability distribution over the process state RVs in time slice 0, i.e. $\Pr\left(\mathbf{X}^{(0)}\right)$. The process state with respect to time may then be represented by the following CPD:

$$\Pr\left(\mathbf{X}^{(t+1)}|\mathbf{X}^{(0)},\ldots,\mathbf{X}^{(t)}\right) \quad \text{for} \quad t>0. \tag{3.1}$$

Note here that $\mathbf{X}^{(t+1)}$ depends on $\{\mathbf{X}^{(0)},\ldots,\mathbf{X}^{(t)}\}$, the entire history of process states before time $t+1$. However, a simplifying assumption, called the Markov assumption[2], is typically assumed to restrict the nature of the dependence of $\mathbf{X}^{(t+1)}$. A first order Markov assumption restricts the nature of this dependence to only $\mathbf{X}^{(t)}$ by assuming that

$$\Pr\left(\mathbf{X}^{(t+1)}|\mathbf{X}^{(0)},\ldots,\mathbf{X}^{(t)}\right) = \Pr\left(\mathbf{X}^{(t+1)}|\mathbf{X}^{(t)}\right). \tag{3.2}$$

This implies that the RVs $\mathbf{X}^{(t+1)}$ are only dependent on the RVs $\mathbf{X}^{(t)}$ for any time slice $t>0$. In other words the future state $\mathbf{X}^{(t+1)}$ is independent of all past states $\mathbf{X}^{(t')}$, where $t'<t$, if the present state $\mathbf{X}^{(t)}$ is known – this type of independence is also known as **conditional independence**.

Typically, it is further assumed that the process being modelled is **stationary**[3]. This means that the CPDs $\Pr\left(\mathbf{X}^{(t'+1)}|\mathbf{X}^{(t')}\right)$ are the same for all time slices $t' \geq 0$, and may be denoted by:

$$\Pr\left(\mathbf{X}^{(t'+1)}|\mathbf{X}^{(t')}\right) \equiv \Pr\left(\mathbf{X}'|\mathbf{X}\right). \tag{3.3}$$

[1] A more comprehensive treatment of DBNs may be found in the textbooks by Barber (2011) and Koller and Friedman (2009).

[2] The Markov assumption facilitates the compact representation of DBNs.

[3] Note that stationary process behaviour typically applies under NOC and that when a fault manifests, process behaviour will typically become non-stationary. Rather, in this thesis, we attempt to model stationary process behaviour under NOC, detect deviation from this behaviour, and identify the cause of deviation given a library of broadly descriptive fault signatures.

This allows for the transition model for process states from one time slice to the next to be conveniently encoded in one CPD, viz. $\Pr\left(\mathbf{X}'|\mathbf{X}\right)$.

A process for which the aforementioned assumptions are valid is referred to as a Markovian process (Dynkin, 1965). For such a process, the state transition CPD, $\Pr\left(\mathbf{X}'|\mathbf{X}\right)$, may be represented by a two-time-slice Bayesian network (Definition 3.1).

Definition 3.1: Two-time-slice Bayesian network (2-TBN)

For a Markovian process whose RVs, representing the time slice t process state, are denoted by $\mathbf{X}^{(t)}$, the CPD $\Pr\left(\mathbf{X}'|\mathbf{X}\right)$ describes the transition of the process state from time slice t' to $t'+1$ for all $t' \geq 0$. Recalling Definition 2.1, this CPD may be represented by a BN. In particular, this BN is referred to as a two-time-slice Bayesian network (2-TBN) since its nodes represent RVs in two time slices, t' and $t'+1$.

Finally, the DBN representing the JPD of process states from time slice 0 to $t+1$, $\Pr\left(\mathbf{X}^{(0)}, \ldots, \mathbf{X}^{(t+1)}\right)$, is defined by Definition 3.2.

Definition 3.2: Dynamic Bayesian network (DBN)

Let $\mathcal{B}_{\rightarrow}$ be a 2-TBN that represents the probability distribution $\Pr\left(\mathbf{X}'|\mathbf{X}\right)$ and let $\mathcal{B}_0$ be a BN that represents the probability distribution $\Pr\left(\mathbf{X}^{(0)}\right)$. A DBN is defined by the pair: $\langle \mathcal{B}_{\rightarrow}, \mathcal{B}_0 \rangle$.

For any time slice $t+1$, $\Pr\left(\mathbf{X}^{(0)}, \ldots, \mathbf{X}^{(t+1)}\right)$ can be computed by:

$$\begin{aligned}\Pr\left(\mathbf{X}^{(0)}, \ldots, \mathbf{X}^{(t+1)}\right) &= \Pr\left(\mathbf{X}^{(0)}\right) \cdot \prod_{t'=0}^{t} \Pr\left(\mathbf{X}^{(t'+1)}|\mathbf{X}^{(t')}\right) \\ &= \Pr\left(\mathbf{X}^{(0)}\right) \cdot \prod_{t'=0}^{t} \Pr\left(\mathbf{X}'|\mathbf{X}\right)\end{aligned}$$

3.1.2 Example: Draining liquid level

Consider again the liquid level example of Figure 2.5 on page 30, but suppose the liquid in the tank is draining slowly over time as shown in Figure 3.1.

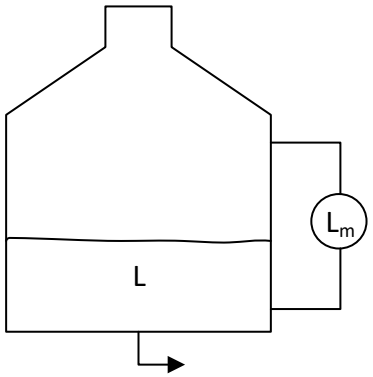


Figure 3.1: Schematic for draining liquid level example. L is the actual level of liquid in the tank and L_m is its measured value.

A DBN may be used to model this process because the actual liquid level decreases over time. Note that the independence structure for all RVs in one time slice is the same as that of the original BN in Figure 2.6 on page 31. Therefore, all the intra time slice (i.e. within a time slice) edges in the 2-TBN, $\mathcal{B}_{\rightarrow}$, and initial BN, $\mathcal{B}_0$, are the same as in Figure 2.6. These edges represent (approximately) the immediate, with respect to time, effect of one RV on another – they are approximate due to differences in process dynamics and the fixed time step progression of a DBN.

The remaining edges in the 2-TBN which need to be added are called inter time slice edges. These edges represent the effect of one RV on another over a period of time (approximately) equal to the difference in real-time between time slices. The liquid level at time slice $t+1$, $L^{(t+1)}$, decreases relative to its value at time slice t, $L^{(t)}$. Therefore, an edge from $L^{(t)}$ to $L^{(t+1)}$ should be created in $\mathcal{B}_{\rightarrow}$ to show this relationship.

Additionally, an edge from $S_L^{(t)}$ to $S_L^{(t+1)}$ should be created in $\mathcal{B}_{\rightarrow}$ to define a 'memory' for the sensor status. This edge is created to encode the persistence of fault conditions as an inertial dependence, namely:

$$\begin{aligned} \text{if} \quad & \Pr\left(S_L^{(t)} = \textit{normal}\right) > \Pr\left(S_L^{(t)} = \textit{faulty}\right), \\ \text{then} \quad & \Pr\left(S_L^{(t+1)} = \textit{normal}\right) > \Pr\left(S_L^{(t+1)} = \textit{faulty}\right), \\ \text{otherwise} \quad & \Pr\left(S_L^{(t+1)} = \textit{normal}\right) < \Pr\left(S_L^{(t+1)} = \textit{faulty}\right). \end{aligned} \tag{3.4}$$

Figure 3.2 shows the structure of the 2-TBN, $\mathcal{B}_{\rightarrow}$, for this example. Note that sensor measurements are considered to be independent over time. Therefore, no edge is drawn from $L_m^{(t)}$ to $L_m^{(t+1)}$. Also note that these models may not necessarily be

exact, but aim to capture enough of the process dynamics to reasonably track process behaviour and hence be useful for fault diagnosis purposes.

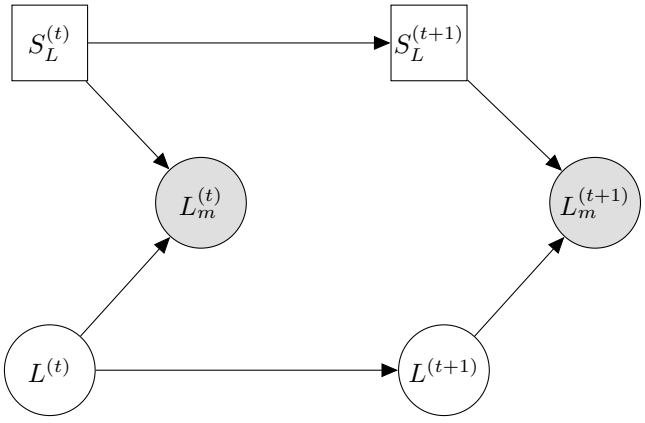


Figure 3.2: Structure of 2-TBN for draining liquid level example. L is the actual level of liquid in the tank, L_m is its measured value, and S_L is the status of the sensor.

3.2 Derivation of DBN inference

Section 3.1 presented the concept of DBNs, stated that they may be used to model process behaviour over time, and introduced a simple example process that may take advantage of DBN modelling. In our case the purpose of the DBN is to facilitate querying the state[4] of the process as a probability distribution over a set of RVs, $\mathbf{X}$, at a particular point in time, $t+1$, given some initial (prior) belief about the state defined by $\mathcal{B}_0$. $\mathbf{X}^{(t+1)}$ consists of all of the process component statuses and PV values (actual and measured) for time slice $t+1$.

There are a few important details to note about $\Pr\left(\mathbf{X}^{(t+1)}\right)$, the first being the time slice $t+1$. This is intentionally specified because the PD approach uses a DBN model to predict process behaviour for time slice $t+1$. The prediction takes the form of a probability distribution for the entire process state, i.e. including probability distributions for measured PVs. Sensor readings taken at the same time slice are then used to infer information about the remaining unobserved RVs in the process state. In particular, the probability distribution over the discrete RVs in $\mathbf{X}$ is of interest for fault diagnosis purposes since these RVs describe the process component statuses.

Furthermore, $\Pr\left(\mathbf{X}^{(t+1)}\right)$ is a hybrid probability distribution therefore it becomes less practical to compute $\Pr\left(\mathbf{X}^{(t+1)}\right)$ as t increases – due to a problem referred to as exponential blow-up. This section first discusses the problem of exponential blow-up

[4]Note that this refers to determining the overall operational mode of the process, i.e. normal or faulty, and the distributions of values for various PVs at a particular point in time.

and then presents an approach to tackle this problem using an inference engine (based on Lerner *et al.* (2000) and Lerner (2002)) tailored to our needs.

3.2.1 Problem of exponential blow-up

Recall from Definition 3.2 that a DBN consists of an initial state defined by $\mathcal{B}_0$ and a transition model defined by $\mathcal{B}_{\rightarrow}$. In order to compute $\Pr\left(\mathbf{X}^{(t+1)}\right)$ one would typically compute the JPD $\Pr\left(\mathbf{X}^{(0)}, \ldots, \mathbf{X}^{(t+1)}\right)$ according to Definition 3.2 and then marginalise out all states not in time slice $t+1$ – additionally, one would typically condition the JPD on all the observations up to time $t+1$ as well. To illustrate, in order to compute $\Pr\left(\mathbf{X}^{(1)}\right)$ one would first compute the JPD $\Pr\left(\mathbf{X}^{(0)}, \mathbf{X}^{(1)}\right)$ and then marginalise out $\mathbf{X}^{(0)}$. Similarly, to compute $\Pr\left(\mathbf{X}^{(2)}\right)$ one would first compute the JPD $\Pr\left(\mathbf{X}^{(0)}, \mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right)$ and then marginalise out $\{\mathbf{X}^{(0)}, \mathbf{X}^{(1)}\}$. This becomes problematic as t increases when $\mathbf{X}$ contains discrete RVs.

When multiplying two independent discrete probability distributions, the number of entries in the resulting probability table is always more than the number of entries in the probability tables of either discrete probability distribution. In fact, in this case, the number of entries in the resulting probability table is the product of the number of entries in the probability tables of each discrete probability distribution.

To illustrate, if there are $t+1$ independent discrete probability distributions whose probability tables each contain m entries each, then the JPD over all involved discrete RVs will be a probability table containing m^{t+1} entries.[5] This probability table becomes exponentially larger as t increases, hence describing the problem which is called **exponential blow-up**. This problem makes exact computation of the JPD intractable as the amount of memory required to store the probability table of the JPD increases exponentially with t. Naturally, this is not practical for an online process monitoring tool.

Note that the above explanation focuses on the multiplication of independent discrete probability distributions. However, there are both discrete and continuous RVs in $\mathbf{X}$, and they have some dependencies between one another. Therefore, the problem of exponential blow-up is further described below – in the context of the PD approach.

Since all continuous RVs are modelled using the family of Gaussian probability distributions in this thesis, $\Pr(\mathbf{X})$ may be represented as a hybrid mixture of Gaussians (see Section 2.3.2). In $\Pr(\mathbf{X})$ each set of assignments to the discrete RVs in $\mathbf{X}$ defines one Gaussian mixture component in $\Pr(\mathbf{X})$ – Table 2.3 illustrates this

[5] It is important to note that we frame this discussion in the context of simple probabilistic methods, and that more advanced methods (such as those presented in Minka (2001)) may reduce or avoid this altogether.

type of probability distribution.

Now suppose that $\Pr\left(\mathbf{X}^{(0)}\right)$ contains m possible sets of assignments to the discrete RVs in $\mathbf{X}$ and therefore m corresponding Gaussian mixture components. To compute $\Pr\left(\mathbf{X}^{(0)}, \mathbf{X}^{(1)}\right)$, one would multiply $\Pr\left(\mathbf{X}^{(0)}\right)$ with $\Pr\left(\mathbf{X}'|\mathbf{X}\right)$, a conditional hybrid probability distribution with m^2 assignments to the discrete RVs in $\{\mathbf{X}, \mathbf{X}'\}$ and m^2 corresponding (conditional) Gaussians over the continuous RVs in $\{\mathbf{X}, \mathbf{X}'\}$. Since $\Pr\left(\mathbf{X}'|\mathbf{X}\right)$ is dependent on $\Pr\left(\mathbf{X}^{(0)}\right)$ the JPD $\Pr\left(\mathbf{X}^{(0)}, \mathbf{X}^{(1)}\right)$ will have m^2 assignments to the discrete RVs in $\{\mathbf{X}^{(0)}, \mathbf{X}^{(1)}\}$ and m^2 corresponding Gaussians over the continuous RVs in $\{\mathbf{X}^{(0)}, \mathbf{X}^{(1)}\}$.

The number of entries (number of assignments to the discrete RVs and corresponding Gaussians over the continuous RVs) in the JPD over the various process states increases m-fold with the number of process states. In other words, if $\Pr\left(\mathbf{X}^{(0)}\right)$ contains m entries, then $\Pr\left(\mathbf{X}^{(0)}, \mathbf{X}^{(1)}\right)$ contains m^2 entries, $\Pr\left(\mathbf{X}^{(0)}, \mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right)$ contains m^3 table entries, and so forth. This is illustrated in Figure 3.3.

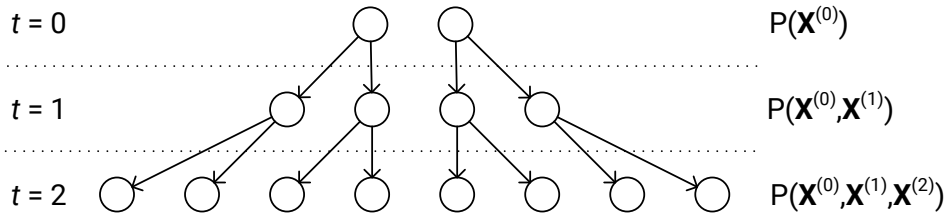


Figure 3.3: Illustration of exponential blow-up when computing a JPD of process states starting an initial state, $\Pr\left(\mathbf{X}^{(0)}\right)$, containing two entries (possible assignments to the discrete RVs in $\mathbf{X}$ and corresponding Gaussians over the continuous RVs in $\mathbf{X}$). Each circle represents one entry in the JPD of process states for a time slice.

An entry contains a probability for the particular process operating mode associated with it, and a probability distribution over the PVs (both actual and measured). Following a particular entry in $\Pr\left(\mathbf{X}^{(0)}\right)$ in Figure 3.3 using the arrows leading from the entry all the way to one in time slice 2, reveals one possible **trajectory** for process behaviour over time (up to time slice 2). Each trajectory describes changes to the process operating mode over time as well as its effects on the probability distributions over the PVs.

Under NOC , the trajectory describing normal process behaviour should usually be the most likely, while trajectories describing multiple process component faults

should be least likely. This means that as the number of trajectories increases (due to time), the number of trajectories with low relative likelihoods also increase, potentially exponentially. Some trajectories may not be worthwhile to track, because of the added computational requirements and inevitably low relative likelihood. In other words, tracking a reduced set of trajectories may not necessarily degrade tracking of process behaviour over time and may allow us to control the problem of exponential blow-up by avoiding further computation on relatively unlikely trajectories. However, this approach is still relatively computationally expensive compared to the one presented in the next section.

3.2.2 Solution: Approximate process state tracking

Up to this point the approach to compute a process state for time slice $t+1$ would always involve the use of an initial state at time slice 0 to compute the JPD over all process states from time slice 0 to $t+1$. However, as described in Section 3.2.1, this creates the problem of exponential blow-up as t increases and results in the tracking of many trajectories describing relatively unlikely process behaviour (these are not particularly useful for tracking actual process behaviour).

A more practical approach, as presented in Lerner *et al.* (2000) and Lerner (2002), would instead use the belief about the process state at time slice t to produce only the set of process state transitions from time slice t to $t+1$, and use those transitions to compute the belief about the process state at time $t+1$. However, doing this results in approximation of the belief about the process state at time slice $t+1$ for all time slices $t > 0$ – since only transitions from the belief about the process state at time slice t are used to compute the belief about the process state at time slice $t+1$ instead of the full set of trajectories up to time $t+1$. This means that several (unlikely) trajectories remain unexplored as t increases.

In theory, not exploring unlikely trajectories is not particularly detrimental to tracking process behaviour because (as described in Section 3.2.1) actual process behaviour over time is typically tracked by the few trajectories which maintain high relative likelihood over time. Therefore, although approximating the belief about the process state at time slice $t+1$ may not be as accurate and reliable as computing the exact belief about the process state for the same time slice, the approximated beliefs about process states over time may still reasonably represent and track actual process behaviour.

However, how closely the approximated beliefs about process states track actual process behaviour depends on how the approximation is performed. Recall that, in order to approximate the time slice $t+1$ belief about the process state from a JPD containing the state RVs at both time slices t and $t+1$, one must marginalise out the state RVs in time slice t from the JPD. Since, in our case, this JPD is represented as a hybrid probability distribution, doing this involves collapsing a hybrid mixture

of Gaussians depending on the assignments to the discrete RVs in time slice $t+1$. Typically, if the mixture components are similar in terms of mean and variance, collapsing the hybrid mixture of Gaussians should not have adverse effects on the process behaviour described by the approximated belief about the process state. On the other hand, if the mixture components are dissimilar, then information loss due to collapsing becomes an issue.

To illustrate, consider a mixture of (univariate) Gaussians with two dissimilar components, i.e. with entirely different means and variances. Given the weighting of each mixture component, the mixture may be collapsed into one Gaussian probability distribution according to Equations 2.36 and 2.37. Figure 3.4 shows such a mixture of Gaussians with two dissimilar mixture components and their collapsed Gaussian probability distribution given equal weighting for each mixture component. Notice that the mean and variance of the collapsed Gaussian probability distribution are considerably different from either mixture component.

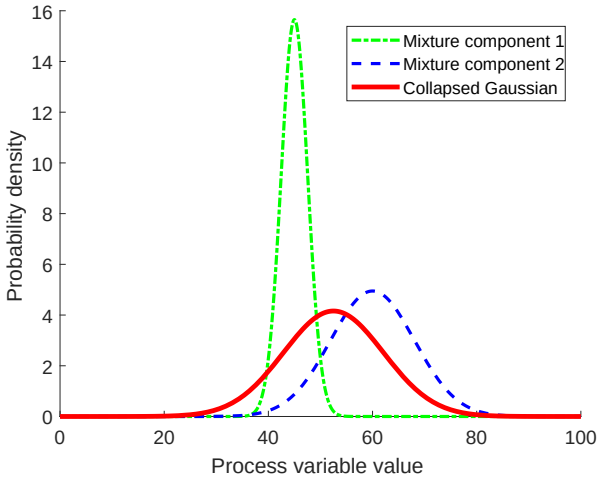


Figure 3.4: Collapsing a mixture of two Gaussians with dissimilar mixture components and equal weighting.

In particular, if the two mixture components describe process behaviour under different operating modes, then the result of collapsing the two mixture components into one Gaussian probability distribution describes entirely different process behaviour. This is further illustrated for the case of the draining liquid level example on page 46. Suppose that the process state for time slice $t+1$ needs to be computed from the JPD:

$$\Pr\left(L^{(t)}, L_m^{(t)}, S_L^{(t)}, L^{(t+1)}, L_m^{(t+1)}, S_L^{(t+1)}\right). \tag{3.5}$$

This requires that all the RVs at time slice t be marginalised out from the JPD. Also suppose that the sensor status, S_L, could be either normal (n) or faulty (f), then this JPD is a hybrid probability distribution containing four Gaussian mixture components,

one for each combination of assignments to $S_L^{(t+1)}$ and $S_L^{(t)}$. Since the approximate belief about the process state at time slice $t+1$ only contains $S_L^{(t+1)}$, this means that the mixture components which depend on the same assignment to $S_L^{(t+1)}$ (for example *n*) need to be collapsed into one Gaussian which is dependent on that same assignment to $S_L^{(t+1)}$. This also means that the collapsing ignores assignments to $S_L^{(t)}$.

To illustrate this using the example, first one would marginalise out $\{L^{(t)}, L_m^{(t)}\}$ from the JPD, then for each assignment to $S_L^{(t+1)}$ one would collapse the relevant mixture components into a Gaussian for each of those assignments. For example, for the case of $S_L^{(t+1)}$=*n*, one would collapse the mixture components

$$\Pr\left(L^{(t+1)}, L_m^{(t+1)} | S_L^{(t+1)} = n, S_L^{(t)} = n\right), \text{ and}$$

$$\Pr\left(L^{(t+1)}, L_m^{(t+1)} | S_L^{(t+1)} = n, S_L^{(t)} = f\right)$$

into the Gaussian:

$$\Pr\left(L^{(t+1)}, L_m^{(t+1)} | S_L^{(t+1)} = n\right). \tag{3.6}$$

Because of this problem Lerner *et al.* (2000) and Lerner (2002) develop an alternative approach for approximating beliefs about process states. In their approach they avoid the problem of exponential blow-up while reducing information loss as a result of collapsing by collapsing only similar Gaussian mixture components when approximating beliefs about process states. The rest of this subsection details such an approach based on Lerner (2002).

First, some notation and terminology need to be established. Recall that all process state RVs are represented by $\mathbf{X}$ – this includes component statuses and PV values. The exact belief about the process state for time slice t is denoted by $\Pr\left(\mathbf{X}^{(t)}\right)$ and its approximate counterpart is denoted by $\tilde{\Pr}\left(\mathbf{X}^{(t)}\right)$ – from this point onward approximated beliefs about process states are exclusively used.

$\mathbf{X}$ is also partitioned into observed (measured) $\mathbf{O}$ and unobserved (queried) $\mathbf{Q}$ RVs. Information about all of the RVs in $\mathbf{O}$ is assumed to be available for each time slice via sensor readings, while the RVs in $\mathbf{Q}$ (i.e. latent RVs) describe underlying details about process states. These details typically need to be inferred and cannot be determined exactly due to lack of sensors. Therefore, a probability distribution over $\mathbf{Q}$ only describes particular beliefs about the RVs in $\mathbf{Q}$ and hence that probability distribution over $\mathbf{Q}$ will be referred to as a **belief state**. In particular, the goal of our inference engine is to infer a probability distribution over $\mathbf{Q}$, given process observations up to time slice $t+1$, i.e. the belief state: $\tilde{\Pr}\left(\mathbf{Q}^{(t+1)} | \mathbf{O}^{(t+1)} = \mathbf{o}^{(t+1)}, \ldots, \mathbf{O}^{(1)} = \mathbf{o}^{(1)}\right)$.

Since both continuous and discrete RVs are in $\mathbf{X}$, $\mathbf{O}$, and $\mathbf{Q}$ it is also convenient

to introduce the subscripts $_\Gamma$ and $_\Delta$ which will denote continuous and discrete RVs subsets respectively. For example $\mathbf{X}_\Delta$ denotes all the discrete RVs in $\mathbf{X}$, while $\mathbf{x}_\Delta$ describes a particular set of assignments which may be assumed by those RVs. Because $\mathbf{X}_\Delta$ describes the statuses of various process components in a process, each unique $\mathbf{x}_\Delta$ also describes different process operating modes which may, for example, correspond to NOC , or a particular sensor bias fault, or multiple simultaneous component faults, and so forth. Furthermore, as a result of the dependency of some RVs in $\mathbf{X}_\Gamma$ on those in $\mathbf{X}_\Delta$, this enables $\mathbf{X}$ to model multiple different process behaviours using a compact structure, i.e. a BN.

Since the approximate belief state for time slice $t+1$ is conditioned on all observations up to time slice $t+1$, one may assume that our inference engine requires, at some point, all observations up to time slice $t+1$. However, note that this requirement may be reduced to just the observations for time slice t and $t+1$. This is achieved by first using $\tilde{\Pr}\left(\mathbf{Q}^{(t)}|\mathbf{O}^{(t)}=\mathbf{o}^{(t)}\right)$ and $\mathbf{O}^{(t)}=\mathbf{o}^{(t)}$ to construct $\tilde{\Pr}\left(\mathbf{X}^{(t)}\right)$, then computing transitions of process behaviour from $\tilde{\Pr}\left(\mathbf{X}^{(t)}\right)$, $\tilde{\Pr}\left(\mathbf{X}^{(t+1)},\mathbf{X}^{(t)}\right)$, and using this as well as the observations for time slice $t+1$ to approximate the belief state for time slice $t+1$. One would then follow this procedure for all $t>0$, starting with $t=1$. As a result, the approximate belief state for time slice $t+1$ would incorporate all past observations. Because of this, note that $\tilde{\Pr}\left(\mathbf{Q}^{(t)}\right)$ and, as a result, $\tilde{\Pr}\left(\mathbf{X}^{(t)}\right)$ subsume all past observations even though this will not be explicitly indicated throughout the rest of this thesis.

As previously discussed, when marginalising out discrete RVs from hybrid probability distributions, some of the Gaussian mixture components need to be collapsed. In our case, $\tilde{\Pr}\left(\mathbf{X}^{(t+1)},\mathbf{X}^{(t)}\right)$ is a hybrid mixture of Gaussians containing the transitions of process behaviour which we require to approximate the belief state at time slice $t+1$. In order to approximate this belief state, we will have to collapse some of the mixture components (Gaussians). However, to reduce the problem of information loss we use an alternative collapsing scheme which collapses the components based on their similarity to one another instead of the assignments of values to discrete RVs on which they are conditioned. Note that this similarity will only apply to the components defined over the RVs in $\tilde{\Pr}\left(\mathbf{X}_\Gamma^{(t+1)}\right)$, since we are interested in approximating the belief state at time slice $t+1$. Further note that we will discuss how similarity is determined after discussing how to represent the structure resulting from this non-traditional collapsing scheme approach. This is necessary because similar components may be conditioned on different assignments of values to $\{\mathbf{X}_\Delta^{(t+1)},\mathbf{X}_\Delta^{(t)}\}$ which means that a traditional hybrid probability distribution cannot be used to represent the result.

To illustrate this, consider the draining liquid level example system on page 46

of Section 3.1.2. Suppose we compute $\tilde{\Pr}\left(\mathbf{X}^{(t+1)}, \mathbf{X}^{(t)}\right)$ and marginalise out $\mathbf{X}_{\Gamma}^{(t)}$ for this system and obtain the hybrid mixture of Gaussians in Table 3.1.

Table 3.1: Illustration of JPD of process states for draining liquid level example. The $\Pr_i$ and $\mathcal{N}_i$ indicate place holders for probability and Gaussian probability distribution respectively.

$\mathbf{X}_{\Delta}^{(t)}$ $S_L^{(t)}$	$\mathbf{X}_{\Delta}^{(t+1)}$ $S_L^{(t+1)}$	$\tilde{\Pr}\left(\mathbf{X}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)}\right)$ $\tilde{\Pr}\left(S_L^{(t)}, S_L^{(t+1)}\right)$	$\tilde{\Pr}\left(\mathbf{X}_{\Gamma}^{(t+1)} \vert \mathbf{X}_{\Delta}^{(t)} = \mathbf{x}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)} = \mathbf{x}_{\Delta}^{(t+1)}\right)$ $\tilde{\Pr}\left(L^{(t+1)}, L_m^{(t+1)} \vert S_L^{(t+1)}, S_L^{(t)}\right)$
n	n	$\Pr_1$	$\mathcal{N}_1$
n	f	$\Pr_2$	$\mathcal{N}_2$
f	n	$\Pr_3$	$\mathcal{N}_3$
f	f	$\Pr_4$	$\mathcal{N}_4$

Traditionally, marginalising out $\mathbf{X}_{\Delta}^{(t)}$ here would result in two components in the hybrid mixture of Gaussians $\tilde{\Pr}\left(\mathbf{X}^{(t+1)}\right)$, one from the collapsing of $\mathcal{N}_1$ and $\mathcal{N}_3$ and the other from the collapsing of $\mathcal{N}_2$ and $\mathcal{N}_4$. Also, the weight of each component in $\tilde{\Pr}\left(\mathbf{X}^{(t+1)}\right)$ would be the sum of the probabilities for each transition involved in the collapsing, i.e. $\Pr_1$ and $\Pr_3$ for $\{\mathcal{N}_1, \mathcal{N}_3\}$ and $\Pr_2$ and $\Pr_4$ for $\{\mathcal{N}_2, \mathcal{N}_4\}$. However, as previously discussed, this approach risks information loss when components being collapsed into one Gaussian are dissimilar.

Suppose $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$ are similar, i.e. in terms of mean and variance, to one another and $\mathcal{N}_4$ is entirely different. The aforementioned traditional collapsing scheme would certainly lose some information when collapsing $\mathcal{N}_2$ and $\mathcal{N}_4$ into one Gaussian. To reduce information loss, our inference engine would instead choose the collapsing scheme which collapses $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$ into one Gaussian and keeps $\mathcal{N}_4$ separate – but how do we store this information and use it for inference for future time slices?

All of the components being collapsed into one Gaussian are similar, but are conditioned on different assignments to $\{\mathbf{X}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)}\}$. Each component being collapsed into one Gaussian must also be weighted for compatibility with Equations 2.36 and 2.37. In the context of our inference engine, this means that the transition, characterised by the particular assignment to $\{\mathbf{X}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)}\}$, also receives the same weighting as its corresponding mixture component. In fact, a probability distribution over $\{\mathbf{X}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)}\}$ can be induced from these weights by assigning a probability of zero to all entries in $\Pr\left(\mathbf{X}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)}\right)$ which cannot be associated with a corresponding weight, and using the appropriate weights to determine the

probabilities of the remaining entries.[6]

To illustrate this, consider collapsing $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$ into one Gaussian. The induced probability distribution over $\{\mathbf{X}_\Delta^{(t)}, \mathbf{X}_\Delta^{(t+1)}\}$ in this case would be:

Table 3.2: Illustration of induced probability distribution over $\{\mathbf{X}_\Delta^{(t)}, \mathbf{X}_\Delta^{(t+1)}\}$ for draining liquid level example when collapsing $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$ into one Gaussian. Note that $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$ are each associated with a particular assignment to $\{\mathbf{X}_\Delta^{(t)}, \mathbf{X}_\Delta^{(t+1)}\}$, thus their weights can each correspond to a particular entry in $\Pr\left(\mathbf{X}_\Delta^{(t)}, \mathbf{X}_\Delta^{(t+1)}\right)$.

$\mathbf{X}_\Delta^{(t)}$	$\mathbf{X}_\Delta^{(t+1)}$	$\Pr\left(\mathbf{X}_\Delta^{(t)}, \mathbf{X}_\Delta^{(t+1)}\right)$
$S_L^{(t)}$	$S_L^{(t+1)}$	$\Pr\left(S_L^{(t)}, S_L^{(t+1)}\right)$
n	n	$\frac{\Pr_1}{\Pr_1+\Pr_2+\Pr_3}$
n	f	$\frac{\Pr_2}{\Pr_1+\Pr_2+\Pr_3}$
f	n	$\frac{\Pr_3}{\Pr_1+\Pr_2+\Pr_3}$
f	f	0

The result of this collapsing thus describes a hybrid probability distribution where every assignment to its discrete RVs describes the same one Gaussian. This probability distribution may be used to represent one hypothesis about the state of the process for time slice $t+1$. The likelihood of this hypothesis is the sum of the likelihoods of each Gaussian component collapsed into the Gaussian of the hypothesis, i.e. for the case of collapsing $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$ into one Gaussian: $\Pr_1 + \Pr_2 + \Pr_3$. Note that this also implies that a different hypothesis with its own likelihood and hybrid probability distribution would exist for $\mathcal{N}_4$.

It is better to further formalise this discussion in terms of $\mathbf{Q}$ since we are primarily interested in the unobserved RVs. Suppose one has the approximate belief state $\tilde{\Pr}\left(\mathbf{Q}^{(t)}|\mathbf{O}^{(t)} = \mathbf{o}^{(t)}, \ldots, \mathbf{O}^{(1)} = \mathbf{o}^{(1)}\right)$ and the observations $\mathbf{o}^{(t)}$ at time slice t, one can then use the 2-TBN to compute

$$\tilde{\Pr}\left(\mathbf{X}^{(t+1)}, \mathbf{Q}^{(t)}|\mathbf{O}^{(t)} = \mathbf{o}^{(t)}, \ldots, \mathbf{O}^{(1)} = \mathbf{o}^{(1)}\right). \tag{3.7}$$

Conditioning on the observations $\mathbf{o}^{(t+1)}$ for time slice $t+1$ and marginalising out $\mathbf{Q}_\Gamma^{(t)}$ yields

$$\tilde{\Pr}\left(\mathbf{Q}^{(t+1)}, \mathbf{Q}_\Delta^{(t)}|\mathbf{O}^{(t+1)} = \mathbf{o}^{(t+1)}, \ldots, \mathbf{O}^{(1)} = \mathbf{o}^{(1)}\right). \tag{3.8}$$

This is a hybrid mixture of Gaussians describing transitions of process behaviour from a given approximate belief about the process state. Each Gaussian describes a

[6] Note that this procedure constructs new a probability distribution and should not be viewed as a modification of another probability distribution.

probability distribution over $\mathbf{Q}_{\Gamma}^{(t+1)}$ and determines which entries in 3.8 (i.e. transitions) should be combined with one another, based on the similarity of their Gaussians, when creating hypotheses. Each **hypothesis** thus consists of a Gaussian over $\mathbf{Q}_{\Gamma}^{(t+1)}$, a probability table over $\mathbf{Q}_{\Delta}^{(t+1)}$ (note we ignore $\mathbf{X}_{\Delta}^{(t)}$ when creating this), and a likelihood.

We also define a BN, separate from the original DBN, which simply stores hypotheses for a time slice t.[7] We refer to this BN as a **belief network**, and denote it as $\mathcal{B}_{\text{bel}}^{(t)}$. $\mathcal{B}_{\text{bel}}^{(t)}$ contains three RVs $H_{\text{bel}}^{(t)}$, $\Delta_{\text{bel}}^{(t)}$, and $\Gamma_{\text{bel}}^{(t)}$, and has the structure illustrated in Figure 3.5.

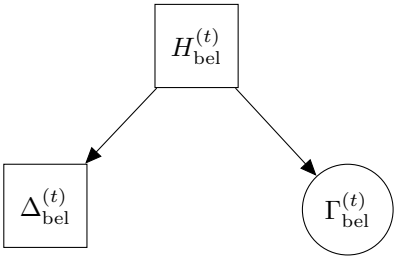


Figure 3.5: Illustration of time slice t belief network structure. $H_{\text{bel}}^{(t)}$ is a discrete RVs used to denote hypotheses and allows for selection of different hypotheses in the belief network. $\Delta_{\text{bel}}^{(t)}$ is a discrete RVs conditioned on H_{bel}, for each assignment to $H_{\text{bel}}^{(t)}$ it contains one discrete probability distribution over the RVs $\mathbf{Q}_{\Delta}$. $\Gamma_{\text{bel}}^{(t)}$ is a continuous RVs conditioned on $H_{\text{bel}}^{(t)}$, for each assignment to $H_{\text{bel}}^{(t)}$ it contains one multivariate Gaussian over the RVs $\mathbf{Q}_{\Gamma}$.

A hypothetical belief state in $\mathcal{B}_{\text{bel}}^{(t)}$ can be accessed by conditioning $H_{\text{bel}}^{(t)}$ on a specific hypothesis $h_{\text{bel}}^{(t)}$. Doing this produces $\Pr\left(\Delta_{\text{bel}}^{(t)}, \Gamma_{\text{bel}}^{(t)} | H_{\text{bel}}^{(t)} = h_{\text{bel}}^{(t)}\right)$, a hybrid probability distribution whose discrete part describes a probability table over $\mathbf{Q}_{\Delta}^{(t)}$ and whose continuous part describes a multivariate Gaussian over $\mathbf{Q}_{\Gamma}^{(t)}$. The likelihood of this hypothetical belief state is $\Pr\left(H_{\text{bel}}^{(t)} = h_{\text{bel}}^{(t)}\right)$.

It is then assumed that one can approximate the time slice t belief state by marginalising out $H_{\text{bel}}^{(t)}$ from $\mathcal{B}_{\text{bel}}^{(t)}$, i.e.

$$\int_{H_{\text{bel}}^{(t)}} \Pr\left(H_{\text{bel}}^{(t)}, \Delta_{\text{bel}}^{(t)}, \Gamma_{\text{bel}}^{(t)}\right) \equiv \tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t)}, \mathbf{Q}_{\Gamma}^{(t)} | \mathbf{O}^{(t)} = \mathbf{o}^{(t)}, \ldots, \mathbf{O}^{(1)} = \mathbf{o}^{(1)}\right). \quad (3.9)$$

One may then apply the approach described on page 55, using this approximate belief state for time slice t and the observations for time slice t and $t+1$, to compute transitions in the same form as 3.8. Recall that each transition has associated with

[7] Note that we introduce another BN here to better explain the concepts in our inference approach since we only use simple probabilistic methods. Future (alternative) iterations of the PD approach may not necessarily require this separate BN.

it a Gaussian over $\mathbf{Q}_{\Gamma}^{(t+1)}$. We next define how to measure similarity between these Gaussians and use this to describe a procedure for combining transitions in Algorithm 1 at the end of this subsection.

For two Gaussians defined over the same set of RVs, the average Kullback-Leibler divergence (Kullback and Leibler, 1951) from the one to the other is used as a measure of similarity. Suppose $\mathcal{N}_1$ and $\mathcal{N}_2$ are two Gaussians defined over the same set of RVs. Assume that $\mathcal{N}_1$ and $\mathcal{N}_2$ have the means $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ and covariances $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$ respectively. Let $D_{KL}\left(\mathcal{N}_1||\mathcal{N}_2\right)$ be the Kullback-Leibler divergence from $\mathcal{N}_1$ to $\mathcal{N}_2$ which is calculated as (Cover and Thomas, 2005):

$$\begin{aligned} D_{KL}\left(\mathcal{N}_1||\mathcal{N}_2\right) =& \frac{1}{2}\left(\operatorname{trace}\left(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1\right) - d\right) \\ &+ \frac{1}{2}\left(\log\frac{\det\left(\boldsymbol{\Sigma}_2\right)}{\det\left(\boldsymbol{\Sigma}_1\right)} + \left(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1\right)^T \boldsymbol{\Sigma}_2^{-1}\left(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1\right)\right) \end{aligned} \tag{3.10}$$

where d is the dimension of the Gaussians, i.e. number of RVs over which the Gaussians are defined.

Note that $D_{KL}\left(\mathcal{N}_1||\mathcal{N}_2\right)$ and $D_{KL}\left(\mathcal{N}_2||\mathcal{N}_1\right)$ are different from one another. Therefore, a similarity metric, J, is defined as the mean of the two, i.e.

$$J\left(\mathcal{N}_1, \mathcal{N}_2\right) = \frac{1}{2}\left(D_{KL}\left(\mathcal{N}_1||\mathcal{N}_2\right) + D_{KL}\left(\mathcal{N}_2||\mathcal{N}_1\right)\right) \tag{3.11}$$

This similarity metric describes how dissimilar two Gaussians are to one another and ranges between zero and infinity. In other words, when J is small then the two Gaussians are more similar, and when J is large then the two Gaussians are more dissimilar. Furthermore, when J is zero then the Gaussians are exactly the same.

In order to determine which transitions have similar Gaussian mixture components, a similarity threshold parameter, c, is defined. This parameter describes a fixed threshold for J beyond which two Gaussians, $\mathcal{N}_i$ and $\mathcal{N}_j$, being compared with one another are considered not similar enough, i.e. $\mathcal{N}_i$ and $\mathcal{N}_j$ are considered similar when $J\left(\mathcal{N}_i, \mathcal{N}_j\right) \leq c$ and dissimilar when $J\left(\mathcal{N}_i, \mathcal{N}_j\right) > c$. This allows the inference engine to find and apply a suitable collapsing scheme which partitions the transitions into disjoint subsets, each of which contains those transitions which have similar Gaussian components. The transitions in each such subset can then be combined to create a hypothesis. These hypotheses may then populate the time slice $t+1$ belief network.

Algorithm 1 describes the procedure for finding and applying this collapsing scheme. By considering the transitions in descending order of likelihood, the algorithm first picks the most likely transition, from a set of unpartitioned transitions, to become the base for a new hypothesis. All remaining unpartitioned transitions which have similar

Gaussian mixture components to this base transition are then also added to that subset and removed from the set of unpartitioned transitions. This is done repeatedly, picking the most likely unpartitioned transition each time, until the set of unpartitioned transitions is empty or, for further optimisation of computational resources, the number of these subsets, and therefore hypotheses, equal K. Therefore, after the K-th subset is populated, the transitions in each subset are combined with one another to create one hypothesis per subset and the rest of the unpartitioned transitions are discarded.

In summary, the procedure requires that the user specify two parameters: c and K. c is the similarity threshold for J in Equation 3.11; if $c = \infty$ then any two Gaussians are considered similar and if $c = 0$ then no two Gaussians are considered similar unless they are exactly the same. K is the maximum number of hypotheses created when combining transitions. If $c = 0$, K hypotheses are created, each corresponding to one of the most likely K transitions, and the remaining transitions are discarded. In contrast if $c = \infty$ then only one hypothesis is created and all transitions are combined into one hypothesis with probability equal to unity.

The reader is referred to Appendix A.1 for a complete example demonstrating the application of this inference engine in the context of fault diagnosis, using the draining liquid level example on page 46 as a case study.

Algorithm 1 Procedure for combining transitions (Lerner, 2002). Note that this procedure constructs various probability distributions, thus some probability distributions may not be legitimate until return.

procedure COMBINETRANSITIONS($\mathcal{A}$,c,K)
  $\mathcal{A} = \langle w_g, \mathcal{N}_g \rangle_{g=1,\ldots,n}$ is a hybrid mixture of n Gaussians representing the transitions in $\tilde{\Pr}\left(\mathbf{Q}_\Delta^{(t)}, \mathbf{Q}^{(t+1)} | \mathbf{O}_\Delta^{(t)} = \mathbf{o}_\Delta^{(t)}, \mathbf{O}^{(t+1)} = \mathbf{o}^{(t+1)}\right)$.
  c is the threshold for the similarity metric J.
  K is the maximum allowed number of hypotheses.
  Let $\mathcal{B}_{\text{bel}}^{(t+1)}$ be an empty belief network containing the RVs $H_{\text{bel}}^{(t+1)}$, $\Delta_{\text{bel}}^{(t+1)}$, and $\Gamma_{\text{bel}}^{(t+1)}$ as in Figure 3.5.
  Sort $\mathcal{A}$ in order of decreasing weights w_g.
  Mark all Gaussians in $\mathcal{A}$ as unused.
  Set $u = 1$
  while $u < K$ and $\mathcal{A}$ contains unused Gaussians **do**
    Let $\langle w, \mathcal{N} \rangle$ be the first unused Gaussian in $\mathcal{A}$.
    Find all indices of Gaussians, $l \in L$, such that $\langle w_l, \mathcal{N}_l \rangle \in \mathcal{A}$ is unused and $J(\mathcal{N}, \mathcal{N}_l) \leq c$ according to Equation 3.11.
    Mark $\langle w, \mathcal{N} \rangle$ and $\langle w_l, \mathcal{N}_l \rangle_{l \in L}$ as used.
    $w' \leftarrow w + \sum_{l \in L} w_l$.
    $\Pr\left(H_{\text{bel}}^{(t+1)} = h_{\text{bel},u}^{(t+1)}\right) \leftarrow w'$
    Divide w and all w_l by w', i.e. normalise.
    Let $\mathcal{N}'$ be the Gaussian resulting from collapsing $\langle w, \mathcal{N} \rangle$ and $\langle w_l, \mathcal{N}_l \rangle_{l \in L}$ according to Equations 2.36 and 2.37.
    $\Pr\left(\Gamma_{\text{bel}}^{(t+1)} | H_{\text{bel}}^{(t+1)} = h_{\text{bel},u}^{(t+1)}\right) \leftarrow \mathcal{N}'$
    for each entry, $\{\mathbf{q}_\Delta^{(t)}, \mathbf{q}_\Delta^{(t+1)}\}$, in $\Pr\left(\mathbf{Q}_\Delta^{(t)}, \mathbf{Q}_\Delta^{(t+1)}\right)$ **do**
      Set $\Pr\left(\mathbf{Q}_\Delta^{(t)} = \mathbf{q}_\Delta^{(t)}, \mathbf{Q}_\Delta^{(t+1)} = \mathbf{q}_\Delta^{(t+1)}\right) \leftarrow 0$
      Find all transitions among $\langle w, \mathcal{N} \rangle$ and $\langle w_l, \mathcal{N}_l \rangle_{l \in L}$ whose discrete RVs assignments match $\{\mathbf{q}_\Delta^{(t)}, \mathbf{q}_\Delta^{(t+1)}\}$ and add their weights to $\Pr\left(\mathbf{Q}_\Delta^{(t)} = \mathbf{q}_\Delta^{(t)}, \mathbf{Q}_\Delta^{(t+1)} = \mathbf{q}_\Delta^{(t+1)}\right)$
    end for
    $\Pr\left(\Delta_{\text{bel}}^{(t+1)} | H_{\text{bel}}^{(t+1)} = h_{\text{bel},u}^{(t+1)}\right) \leftarrow \int_{\mathbf{Q}_\Delta^{(t)}} \Pr\left(\mathbf{Q}_\Delta^{(t)}, \mathbf{Q}_\Delta^{(t+1)}\right)$
    Increment u.
  end while
  return $\mathcal{B}_{\text{bel}}^{(t+1)}$
end procedure

3.2.3 Fault detection and root cause analysis

Since the inference engine produces a probability distribution over all possible discrete RVs (process component statuses), one can easily analyse this distribution to learn more about process abnormalities and their possible root causes. In order to determine process abnormality, an **Abnormality Likelihood Index**[8] (ALI) (Yu and Rashid, 2013) is introduced here. This index simply considers the probability of all operating modes which are abnormal, i.e. for a time slice t:

$$\text{ALI}^{(t)} = 1 - \Pr\left(\mathbf{X}_{\Delta}^{(t)} = \mathbf{x}_{\Delta,NOC}^{(t)}\right) \tag{3.12}$$

where $\mathbf{x}_{\Delta,NOC}^{(t)}$ is the particular assignment (at time slice t) to the set of discrete RVs which represents NOC. Typically this will be the first entry in the table for $\Pr\left(\mathbf{X}_{\Delta}^{(t)}\right)$. The case study in Appendix A.1 is extended in Appendix A.2 to show tracking of the process state over time as well as the usage of ALI values over time. The user may also choose a particular threshold for ALI over time, denoted as $\text{ALI}_{\text{limit}}$. For time slice t, when $\text{ALI}^{(t)} > \text{ALI}_{\text{limit}}$ the process is considered to be behaving abnormally.

Once abnormal process behaviour is detected, root cause analysis follows a similar procedure by considering the ALI of each individual component instead of the overall ALI, i.e. for a component $C \in \mathbf{X}_{\Delta}$ at time slice t:

$$\text{ALI}_{C}^{(t)} = 1 - \Pr\left(C^{(t)} = c_{NOC}^{(t)}\right) \tag{3.13}$$

where $c_{NOC}^{(t)}$ is the particular assignment (at time slice t) to C which represents the *normal* status. This will also typically be the first entry in $\Pr\left(C^{(t)}\right)$. After computing the ALIs for each component, one can then compare those ALIs with one another and identify the components with high ALIs as possible fault causes.

Ideally, for a single fault cause, the component ALI of the root cause should be considerably higher than the ALIs of all other components for each sample reported as being abnormal. Averaging these component ALIs for all samples reported as being abnormal and plotting the averages on a bar plot, produces a component ALI plot which can be used to quickly identify fault causes in a similar manner to the relative contribution chart in Figure 2.2 on page 17.

[8] Note that this concept was not included in Lerner *et al.* (2000) or Lerner (2002).

3.3 Computational enhancements to inference engine

The approximation of process states in the inference engine overcomes the problem of exponential blow-up and enables inference to be performed online with limited computational resources. In order to further reduce computational burden, Lerner (2002) describes two enhancements which are compatible with this approach. The first, transition enumeration, can be used to decrease the computational time needed to process new observations and the second, subsystem decomposition, can be used to reduce the amount of memory required to store hypotheses, which may also facilitate storing more hypotheses. These are detailed in this section and used in our PD approach.

3.3.1 Transition enumeration

Recall that the procedure for combining transitions requires the hybrid mixture of Gaussians $\tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t)}, \mathbf{Q}^{(t+1)} | \mathbf{O}_{\Delta}^{(t)} = \mathbf{o}_{\Delta}^{(t)}, \mathbf{O}^{(t+1)} = \mathbf{o}^{(t+1)}\right)$. Each entry in this mixture describes, by way of $\{\mathbf{x}_{\Delta}^{(t+1)}, \mathbf{x}_{\Delta}^{(t)}\}$, a particular change of operational mode from time slice t to $t+1$, and has associated with it a likelihood as well as a Gaussian over $\mathbf{Q}_{\Gamma}^{(t+1)}$ – we will refer to each entry as a transition. As described in the previous section, we combine transitions based on the similarity of their Gaussians. This partitions the transitions into disjoint subsets which can be used to create hypotheses and in turn populate a belief network.

When creating a hypothesis from a disjoint subset containing more than one transition, transitions which have relatively low likelihoods typically contribute less to determining the attributes of that hypothesis. This is a consequence of using a weighted combination of the relevant attributes those transitions to create the probability table over $\mathbf{Q}_{\Delta}$ and the Gaussian over $\mathbf{Q}_{\Gamma}$ associated with that hypothesis. Furthermore, the likelihood of the hypothesis is the sum of the likelihoods of those transitions. This also means that when creating a hypothesis from a disjoint subset containing only one transition which has a relatively low likelihood, that hypothesis will also have a relatively low likelihood. Since a belief state can be approximated as a weighted combination of hypotheses, hypotheses which have relatively low likelihoods will typically also contribute less to determining the attributes of that approximate belief state.

Because of the particular design of the DBNs in this thesis, we can choose which of the transitions in $\tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t)}, \mathbf{Q}^{(t+1)} | \mathbf{O}_{\Delta}^{(t)} = \mathbf{o}_{\Delta}^{(t)}, \mathbf{O}^{(t+1)} = \mathbf{o}^{(t+1)}\right)$ to enumerate instead of computing all of them. This is advantageous since the creation of a hypothesis does not necessarily require all transitions to be enumerated, and furthermore all remaining unpartitioned transitions are discarded at the end of

Algorithm 1. Instead of enumerating all transitions, only enumerating select transitions in $\tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t)}, \mathbf{Q}^{(t+1)} | \mathbf{O}_{\Delta}^{(t)} = \mathbf{o}_{\Delta}^{(t)}, \mathbf{O}^{(t+1)} = \mathbf{o}^{(t+1)}\right)$ may result in the creation of hypotheses with comparatively similar attributes. Thus we may still obtain a similar approximation for the time slice $t+1$ belief state, while reducing computational requirements.

The DBN designs used in this thesis are restricted such that continuous RVs do not have children which are discrete RVs, i.e. no augmented CPDs. Because of this, one can condition a DBN on a particular set of assignments to $\{\mathbf{X}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)}\}$ and compute a Gaussian over $\mathbf{X}_{\Gamma}^{(t+1)}$ associated with this particular set of assignments. This Gaussian may then be used to determine the likelihood of the time slice $t+1$ observations $\mathbf{o}_{\Gamma}^{(t+1)}$ under the process operating mode changes described by the given transition. Once the likelihood of the observed process behaviour is known, it is also possible to compute the posterior likelihood of the transition by multiplying the prior probability of the transition by that likelihood.

Since the posterior likelihood of a transition is dependent on the prior probability of that transition, it is less likely that a transition which has a low prior probability will have a relatively high posterior likelihood. This strongly applies to transitions which describe simultaneous abnormalities for multiple components – these events are typically unlikely *a priori* and do not become more likely *a posteriori*. In particular, we wish to avoid enumerating these transitions which have low prior probabilities, and in turn relatively low posterior likelihoods, since they typically have little impact on the approximation of the belief state.

We refer to the procedure for doing this as **transition enumeration**. It consists of the four key steps listed below. They are subsequently discussed in more depth. Note that transitions which are not enumerated will have zero likelihood in $\tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t)}, \mathbf{Q}^{(t+1)} | \mathbf{O}_{\Delta}^{(t)} = \mathbf{o}_{\Delta}^{(t)}, \mathbf{O}^{(t+1)} = \mathbf{o}^{(t+1)}\right)$ and can be discarded, with limited repercussions, when applying Algorithm 1 after transition enumeration.

1. Computing the prior transition probabilities, $\tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)} | \mathbf{O}_{\Delta}^{(t)} = \mathbf{o}_{\Delta}^{(t)}\right)$.
2. Finding all transitions which are eligible for enumeration. Also note the introduction of two parameters $T_{\max}$, and $\Pr_T$ which we define there.
3. Computing a Gaussian over $\mathbf{X}_{\Gamma}^{(t+1)}$ for each transition that was deemed eligible for enumeration.
4. Computing a posterior likelihood for each transition that was deemed eligible for enumeration.

Computing $\tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)} | \mathbf{O}_{\Delta}^{(t)} = \mathbf{o}_{\Delta}^{(t)}\right)$:

The prior transition probabilities are relatively straightforward to compute. As in

Section 3.2.2, one would use the approximate belief state at time slice t and the observations at time slice t to create an approximate belief about the process state at time slice t which can be propagated through a 2-TBN to produce all of the transitions. However, a key difference here is that all continuous RVs as well as observations for time slice $t+1$ are temporarily ignored. In other words, only the discrete part of the approximate belief about the process state at time slice t is propagated through the 2-TBNresulting in $\tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)} | \mathbf{O}_{\Delta}^{(t)} = \mathbf{o}_{\Delta}^{(t)}\right)$. Each entry in this probability table is the prior probability for a transition.

Finding all transitions which are eligible for enumeration:
As previously discussed, we do not wish to enumerate transitions which have low prior probabilities. Thus we choose the top T transitions with the highest prior probabilities and mark these as eligible for enumeration. The user may specify constraints to restrict T to some maximum or sufficient value, both of which reduce the computational requirements of the inference engine.

The first constraint, a maximal value for T, is easy to impose by simply selecting the top $T = T_{\max}$ transitions from the sorted $\tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)} | \mathbf{O}_{\Delta}^{(t)} = \mathbf{o}_{\Delta}^{(t)}\right)$, where $T_{\max}$ is the maximum number of transitions which may be deemed eligible for further enumeration. For large processes, this is useful for limiting the computational requirements of the inference engine to some absolute maximum such that the computational time for processing new observations is reasonable.

The second constraint, a sufficient value for T, exploits the idea that some transitions describe the occurrence of relatively unlikely events, such as simultaneous abnormalities for multiple components, and have low priori probability and as a result would have relatively low posterior likelihood. This value for T should be just large enough to be able to reasonably approximate the time slice $t+1$ belief state and should not be fixed in each time slice. The second constraint is imposed by choosing T such that the cumulative prior probabilities for the T transitions is at least Pr_T.

Preferably a combination of the two constraints should be used such that one only enumerates $T_{\max}$ transitions when this is necessary, otherwise one should enumerate a sufficient number of $T < T_{\max}$ transitions such that their cumulative prior probabilities is at least Pr_T. Once the T transitions, deemed eligible for enumeration, have been found, their associated Gaussians and posterior likelihoods can be computed.

Computing a Gaussian for each eligible transition:
A Gaussian for a eligible transition can be computed by choosing one eligible transition and considering only the continuous RVs in the approximate belief about the process state and 2-TBN which depend on the particular set of assignments to $\{\mathbf{X}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)}\}$

obtained from that transition. Doing this produces the Gaussian

$$\tilde{\Pr}\left(\mathbf{Q}_{\Gamma}^{(t)}, \mathbf{X}_{\Gamma}^{(t+1)} | \mathbf{O}_{\Gamma}^{(t)}{=}\mathbf{o}_{\Gamma}^{(t)}, \mathbf{X}_{\Delta}^{(t)} = \mathbf{x}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)} = \mathbf{x}_{\Delta}^{(t+1)}\right) \tag{3.14}$$

where $\mathbf{x}_{\Delta}^{(t)}$ and $\mathbf{x}_{\Delta}^{(t+1)}$ are determined from the chosen transition. Note that we will denote $\left(\mathbf{X}_{\Delta}^{(t)} = \mathbf{x}_{\Delta}^{(t)}, \mathbf{X}_{\Delta}^{(t+1)} = \mathbf{x}_{\Delta}^{(t+1)}\right)$ with the shorthand τ_i hereafter.

The next step is to marginalise out $\mathbf{Q}_{\Gamma}^{(t)}$ from 3.14 since these RVs are no longer required. This results in

$$\tilde{\Pr}\left(\mathbf{X}_{\Gamma}^{(t+1)} | \mathbf{O}_{\Gamma}^{(t)}{=}\mathbf{o}_{\Gamma}^{(t)}, \tau_i\right). \tag{3.15}$$

Then, it is left to determine (1) the likelihood of the time slice $t+1$ observations for the continuous RVs and (2) the probability distribution over $\mathbf{Q}_{\Gamma}^{(t+1)}$ given those observations. The first part is done by marginalising out $\mathbf{Q}_{\Gamma}^{(t+1)}$ from 3.15, then using Equation 2.35 to determine the likelihood of the observations $\mathbf{o}_{\Gamma}^{(t+1)}$:

$$\tilde{\Pr}\left(\mathbf{O}_{\Gamma}^{(t+1)} = \mathbf{o}_{\Gamma}^{(t+1)} | \mathbf{O}_{\Gamma}^{(t)}{=}\mathbf{o}_{\Gamma}^{(t)}, \tau_i\right). \tag{3.16}$$

The second part is done by conditioning 3.15 on the observations $\mathbf{o}_{\Gamma}^{(t+1)}$ using Equations 2.44 and 2.45, resulting in

$$\tilde{\Pr}\left(\mathbf{Q}_{\Gamma}^{(t+1)} | \mathbf{O}_{\Gamma}^{(t+1)} = \mathbf{o}_{\Gamma}^{(t+1)}, \mathbf{O}_{\Gamma}^{(t)}{=}\mathbf{o}_{\Gamma}^{(t)}, \tau_i\right). \tag{3.17}$$

Following this procedure for each eligible τ_i produces a likelihood of the time slice $t+1$ observations for the continuous RVs and a Gaussian over $\mathbf{Q}_{\Gamma}^{(t+1)}$ for each eligible transition.

Computing an *a posteriori* probability for each eligible transition:
The posterior likelihood of each eligible transition, τ_i, is computed by multiplying its prior probability by the likelihood of the time slice $t+1$ observations for the continuous RVs as calculated above, i.e.

$$\tilde{\Pr}\left(\tau_i\right) \times \tilde{\Pr}\left(\mathbf{O}_{\Gamma}^{(t+1)} = \mathbf{o}_{\Gamma}^{(t+1)} | \mathbf{O}_{\Gamma}^{(t)}{=}\mathbf{o}_{\Gamma}^{(t)}, \tau_i\right). \tag{3.18}$$

Note that the posterior likelihoods for all eligible transitions will not sum to unity because the eligible transitions are subset of all possible transitions given an approximate belief about a process state. Therefore, either (1) the posterior likelihoods of the eligible transitions need to be normalised or (2) the likelihoods of the hypotheses created from them need to be normalised in order to ensure that the hypotheses in the time slice $t+1$ belief network has a legitimate probability distribution.[9]

[9] Note that normalisation here refers to dividing the elements in a set of values by the sum of those values.

In the first case, one simply sums all the posterior likelihoods of the eligible transitions and divide each by the total before applying Algorithm 1. In the second case, one would apply Algorithm 1 directly to the enumerated set of eligible transitions, i.e. using their posterior likelihoods. Doing this means that the likelihoods of hypotheses in the time slice $t+1$ belief network will not sum to unity and hence need to be normalised.

In this thesis method (1) was used to avoid added complexity in calculations when also using the subsystem decomposition enhancement.

3.3.2 Subsystem decomposition

Recall that each hypothesis h_{bel} in a belief network consists of a hypothesis probability $\Pr\left(H_{\text{bel}} = h_{\text{bel}}\right)$, a discrete probability distribution over $\mathbf{Q}_{\Delta}\ \Pr\left(\Delta_{\text{bel}}|H_{\text{bel}} = h_{\text{bel}}\right)$, and a Gaussian over $\mathbf{Q}_{\Gamma}\ \Pr\left(\Gamma_{\text{bel}}|H_{\text{bel}} = h_{\text{bel}}\right)$. A concern for large processes is that the number of entries in the discrete probability distributions may be very large, causing memory issues with respect to the storage of multiple hypotheses in a belief network. This is because the size of the discrete probability distribution grows exponentially as one monitors more process components. These large discrete probability distribution tables are also typically less practical because the larger these distributions are, the more zero probability entries they will contain – due to those entries typically describing abnormalities for multiple components simultaneously.[10]

A convenient solution to this problem is to decompose the belief network into smaller subsystems which represent individual unit processes – such as reactors, tanks, columns, and so on. Each of these subsystems may then maintain their own hypotheses which contain smaller sized discrete and continuous probability distributions over only the RVs associated with those subsystems. This enables lower storage requirements for storing hypotheses versus a non-decomposed belief network, allowing a decomposed belief network to store more hypotheses on the same amount of storage. Conveniently, these subsystem hypotheses may also be used to produce more detailed process monitoring solutions such as subsystem abnormality detection instead of overall process abnormality detection.[11]

Introducing decomposed belief networks:

Some modifications need to be made to our inference engine so that it may accommodate a decomposed belief network. However, before presenting these modifications, the concept of a decomposed belief network will be further detailed.

[10] Note that this is not necessarily a major problem as one may work with sparse table representations. However, there are drawbacks to such representations, in particular the requirement that a majority of table entries must be equal to zero. Thus sparse table representations will not work for cases such as initial hypotheses where a majority of entries are non-zero.

[11] Since the aim of this thesis is to contrast a model-based PD approach with a traditional MSPM approach, only overall process abnormality will be monitored however.

The concept of a decomposed belief network generalises the non-decomposed belief network (as illustrated in Figure 3.5). Three RVs corresponding to $H_{\text{bel}}^{(t)}$, $\Delta_{\text{bel}}^{(t)}$, and $\Gamma_{\text{bel}}^{(t)}$ used to describe hypotheses for the entire process, are also used in a similar manner to describe hypotheses for each subsystem in a decomposed belief network. Consider a decomposed belief network for time slice t, where the hypothesis RV for a subsystem i is denoted as $Hi_{\text{bel}}^{(t)}$. $\Delta i_{\text{bel}}^{(t)}$ and $\Gamma i_{\text{bel}}^{(t)}$ respectively denote the discrete and continuous probability distributions associated with $Hi_{\text{bel}}^{(t)}$, allowing for selection of different hypotheses for subsystem i by conditioning on $Hi_{\text{bel}}^{(t)}$. For each subsystem hypothesis, $\Delta i_{\text{bel}}^{(t)}$ contains one discrete probability distribution over the discrete RVs in subsystem i which are denoted as $\mathbf{Q}i_{\Delta}^{(t)} \subset \mathbf{Q}_{\Delta}^{(t)}$. Similarly, for each subsystem hypothesis, $\Gamma i_{\text{bel}}^{(t)}$ contains one multivariate Gaussian over the continuous RVs in subsystem i which are denoted as $\mathbf{Q}i_{\Gamma}^{(t)} \subset \mathbf{Q}_{\Gamma}^{(t)}$. Furthermore, the $\mathbf{Q}i_{\Delta}$ form a partition of $\mathbf{Q}_{\Delta}$ and similarly the $\mathbf{Q}i_{\Gamma}$ form a partition of $\mathbf{Q}_{\Gamma}$.

It is also important to note that these subsystems are not necessarily independent of one another, but will have certain subsystem dependence conditions imposed on their dependencies, namely:

1. The discrete part (Δi_{bel}) of one subsystem may only be dependent on the discrete parts of other subsystems, and the continuous parts ($\mathbf{\Gamma} i_{\text{bel}}$) of one subsystem may only be dependent on the continuous parts of other subsystems.

2. Neither discrete nor continuous parts of any subsystem may directly depend on the hypothesis RV of another subsystem.

3. The hypothesis RV of any subsystem may not depend on the hypothesis RV of another subsystem.

4. All dependencies in a belief network may not form cyclic structures in that belief network.

Figure 3.6 illustrates a decomposed belief network for time slice t containing three subsystems.

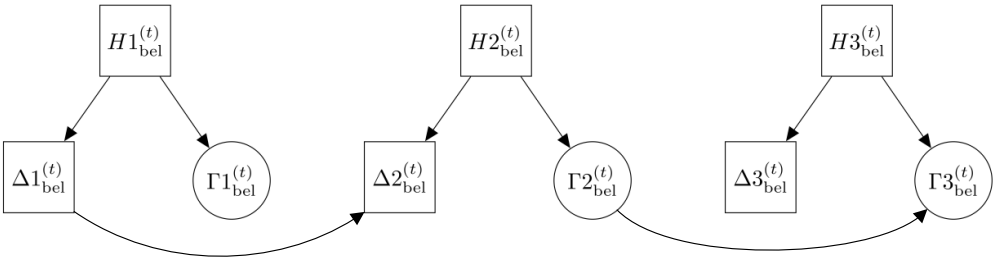


Figure 3.6: Illustration of a decomposed belief network containing three subsystems with some subsystem dependencies. Each subsystem $i \in \{1, 2, 3\}$ stores its hypotheses using the RVs, $Hi^{(t)}_{\text{bel}}$, $\Delta i^{(t)}_{\text{bel}}$, and $\Gamma i^{(t)}_{\text{bel}}$, associated with that subsystem. Also notice that the subsystem dependence conditions are obeyed in this network.

Using decomposed belief networks to compute transitions:

It is now left to discuss the usage and population of a decomposed belief network. In other words, given a decomposed belief network for time slice t, how does one compute transitions of subsystem belief states from time slice t to $t+1$ and use these transitions to populate a decomposed belief network for time slice $t+1$?

The first part can be done in one of two ways (both of which will affect the second part): either one can compute transitions for each subsystem (i.e. track the behaviour of process subsystems rather than overall process behaviour), or one can compute transitions using the approximate belief state over the entire process as has been done previously. In order to use the first approach one must create subsystem DBNs from the entire process DBN and use the inference engine on those to create subsystem hypotheses. However, Lerner (2002) notes that doing this typically produces subsystem DBNs which are similar in size (number of nodes) to that of the entire process DBN. This happens due to the interconnected nature of processes and is required so that the influence of all process observations are accounted for in those transitions, i.e. including observations associated with other subsystems. Therefore, this thesis uses the second approach. This involves computing an approximate belief state using a decomposed belief network, and can be done, in part, by marginalising out all of the Hi_{bel} in that decomposed belief network.

Note that using this approach creates a probability distribution for each Δi_{bel} and each Γi_{bel}. However, the approximate belief state is in the form $\tilde{\Pr}\left(\mathbf{Q}^{(t)}_{\Delta}, \mathbf{Q}^{(t)}_{\Gamma} | \mathbf{O}^{(t)} = \mathbf{o}^{(t)}, \ldots, \mathbf{O}^{(1)} = \mathbf{o}^{(1)}\right)$, thus one needs to construct this probability distribution. In this thesis, we do this by first computing a marginal probability distribution for each RVs in $\mathbf{Q}$ using the probability distributions obtained from the decomposed belief network, after marginalising out all

Hi_{bel}.[12] These marginal probability distributions are then used to define the CPDs in a BN, $\mathcal{B}_t$, which has the same structure as the initial BN $\mathcal{B}_0$. Doing this allows one to use $\mathcal{B}_t$ to approximate the time slice t belief state in the form $\tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t)}, \mathbf{Q}_{\Gamma}^{(t)} | \mathbf{O}^{(t)} = \mathbf{o}^{(t)}, \ldots, \mathbf{O}^{(1)} = \mathbf{o}^{(1)}\right)$.

Modifying procedure for combining transitions:
Using the approximate belief state for time slice t, one can generate transitions between full system states in the same way as described in Section 3.2.2 or Section 3.3.1. These transitions will then need to be combined to create subsystem hypotheses to populate the time slice $t+1$ decomposed belief network. Because some subsystems are dependent on other subsystems some important modifications need to be made to the procedure for combining transitions in Algorithm 1. In order to present these modifications, we introduce the concept of RVs relevant to a subsystem.

We shall use $\mathcal{R}i$ to denote the set of RVs relevant to subsystem i. $\mathcal{R}i$ is defined as all the RVs, discrete and continuous, in subsystem i as well as their direct parent RVs in the decomposed belief network, excluding any hypothesis RVs. $\mathcal{R}i$ is partitioned into two sets, $\mathcal{R}i^{+}$ and $\mathcal{R}i^{-}$, which denote those RVs in and outside subsystem i respectively. Furthermore, the subscripts $_{\Delta}$ and $_{\Gamma}$ are also compatible with $\mathcal{R}i$, $\mathcal{R}i^{+}$, and $\mathcal{R}i^{-}$ and once again denote discrete and continuous RVs subsets.

To illustrate this concept, consider subsystem 2 in Figure 3.6. Excluding any hypothesis RVs, the RVs relevant to subsystem 2 are those associated with subsystem 2, i.e. those in $\Delta 2_{\text{bel}}$ and $\Gamma 2_{\text{bel}}$, and those which subsystem 2 is dependent on, i.e. those in $\Delta 1_{\text{bel}}$. Table 3.3 further illustrates how this set of relevant RVs are distributed into the various subsets of $\mathcal{R}2$.

Table 3.3: Illustration of the concept of RVs relevant to a subsystem for subsystem 2 in Figure 3.6.

$$\mathcal{R}2 = \{\Delta 2_{\text{bel}}, \Gamma 2_{\text{bel}}, \Delta 1_{\text{bel}}\} = \{\mathbf{Q}2_{\Delta}, \mathbf{Q}2_{\Gamma}, \mathbf{Q}1_{\Delta}\}$$

Discrete parts	Continuous parts
$\mathcal{R}2_{\Delta} = \{\Delta 2_{\text{bel}}, \Delta 1_{\text{bel}}\} = \{\mathbf{Q}2_{\Delta}, \mathbf{Q}1_{\Delta}\}$	$\mathcal{R}2_{\Gamma} = \Gamma 2_{\text{bel}} = \mathbf{Q}2_{\Gamma}$
$\mathcal{R}2_{\Delta}^{+} = \Delta 2_{\text{bel}} = \mathbf{Q}2_{\Delta}$	$\mathcal{R}2_{\Gamma}^{+} = \Gamma 2_{\text{bel}} = \mathbf{Q}2_{\Gamma}$
$\mathcal{R}2_{\Delta}^{-} = \Delta 1_{\text{bel}} = \mathbf{Q}1_{\Delta}$	$\mathcal{R}2_{\Gamma}^{-} = \emptyset = \emptyset$

Note that because of the dependencies of some subsystems on other subsystems, some subsystem hypothesis probability distributions have to be stored as CPDs. Subsystem 2 is an example of such a case: for a hypothesis in this subsystem, the probability distribution over $\mathbf{Q}2_{\Delta}$ stored in $\Delta 2_{\text{bel}}$ is conditional on $\mathbf{Q}1_{\Delta}$. This means that if $\mathcal{R}i^{-} = \emptyset$, then Δi_{bel} and Γi_{bel} have discrete probability distributions and Gaussians over $\mathcal{R}i_{\Delta}^{+}$ and $\mathcal{R}i_{\Gamma}^{+}$ respectively, for each subsystem i. On the other hand, if

[12] Note these marginal probability distributions subsume all observations up to time slice t.

$\mathcal{R}i_{\Delta}^{-} \neq \emptyset$ then Δi_{bel} has discrete CPDs over $\mathcal{R}i_{\Delta}^{+}$ given $\mathcal{R}i_{\Delta}^{-}$, and similarly if $\mathcal{R}i_{\Gamma}^{-} \neq \emptyset$ then Γi_{bel} has conditional Gaussians over $\mathcal{R}i_{\Gamma}^{+}$ given $\mathcal{R}i_{\Gamma}^{-}$.

Given a set of transitions, the procedure for combining them to create hypotheses needs to consider each subsystem separately. After finding a suitable collapsing scheme for a subsystem, it is easy to create hypothesis probability distributions in the appropriate form for that subsystem because the transitions describe JPDs over the entire process. This simply requires that some marginalisation and conditioning be performed after combining transitions such that subsystem hypothesis probability distributions are in the appropriate form for each subsystem.

Although the same collapsing scheme can theoretically be used for each subsystem, it is more valuable to use different collapsing schemes in each subsystem as implied above. This enables each subsystem to maintain its own unique hypotheses which better define a belief state describing process behaviour for that subsystem. As a result, when transitions are uninformative about the behaviour of RVs associated with that subsystem, then more hypotheses can be maintained for that subsystem while other subsystems may maintain fewer hypotheses. In order to use different collapsing schemes in each subsystem, the metric for determining the similarity between transitions must be dependent on the RVs relevant to a subsystem. In particular, for a subsystem i, instead of comparing the similarities of the Gaussians associated with each transition, one must first marginalise and condition those such that they are defined over $\mathcal{R}i_{\Gamma}^{+}$ given $\mathcal{R}i_{\Gamma}^{-}$ before comparing them.

This means that when $\mathcal{R}i_{\Gamma}^{-} \neq \emptyset$ then one would compare similarity between conditional Gaussians.[13] The conditional Kullback-Leibler divergence from one Gaussian, $\mathcal{N}_1\left(\mathcal{R}i_{\Gamma}^{+}|\mathcal{R}i_{\Gamma}^{-}\right)$, to another, $\mathcal{N}_2\left(\mathcal{R}i_{\Gamma}^{+}|\mathcal{R}i_{\Gamma}^{-}\right)$, is calculated as (Cover and Thomas, 2005):

$$\begin{aligned} &D_{KL}\left(\mathcal{N}_1\left(\mathcal{R}i_{\Gamma}^{+}|\mathcal{R}i_{\Gamma}^{-}\right)||\mathcal{N}_2\left(\mathcal{R}i_{\Gamma}^{+}|\mathcal{R}i_{\Gamma}^{-}\right)\right) \\ &= D_{KL}\left(\mathcal{N}_1\left(\mathcal{R}i_{\Gamma}^{+},\mathcal{R}i_{\Gamma}^{-}\right)||\mathcal{N}_2\left(\mathcal{R}i_{\Gamma}^{+},\mathcal{R}i_{\Gamma}^{-}\right)\right) - D_{KL}\left(\mathcal{N}_1\left(\mathcal{R}i_{\Gamma}^{-}\right)||\mathcal{N}_2\left(\mathcal{R}i_{\Gamma}^{-}\right)\right). \end{aligned} \tag{3.19}$$

The metric for determining similarity between two transitions for subsystem i is then defined as:

$$\begin{aligned} Ji\left(\mathcal{N}_1,\mathcal{N}_2\right) =& \frac{1}{2}D_{KL}\left(\mathcal{N}_1\left(\mathcal{R}i_{\Gamma}^{+}|\mathcal{R}i_{\Gamma}^{-}\right)||\mathcal{N}_2\left(\mathcal{R}i_{\Gamma}^{+}|\mathcal{R}i_{\Gamma}^{-}\right)\right) \\ &+ \frac{1}{2}D_{KL}\left(\mathcal{N}_2\left(\mathcal{R}i_{\Gamma}^{+}|\mathcal{R}i_{\Gamma}^{-}\right)||\mathcal{N}_1\left(\mathcal{R}i_{\Gamma}^{+}|\mathcal{R}i_{\Gamma}^{-}\right)\right)\Big) \end{aligned} \tag{3.20}$$

Algorithm 2 summarises the modified procedure for combining transitions which

[13] Note that Equation 3.19 is still useful when $\mathcal{R}i_{\Gamma}^{-} = \emptyset$ since the Kullback-Leibler divergence for a Gaussian over an empty set is zero.

results in a decomposed belief network. The procedure still requires that the user specify two parameters, c and K, where c is the similarity threshold for Ji in Equation 3.20 and K is the maximum number of hypotheses that may be created in each subsystem.[14] If $c = 0$, K hypotheses are created per subsystem, each corresponding to one of the most likely K transitions, and the remaining transitions are discarded. In contrast if $c = \infty$ then only one hypothesis is created per subsystem and all transitions are combined into one hypothesis with probability equal to unity.

[14] In practice, one may also define different values of c and K for each subsystem.

Algorithm 2 Subsystem decomposition modified procedure for combining transitions (Lerner, 2002). Note that this procedure constructs various probability distributions, thus some probability distributions may not be legitimate until return.

procedure MODIFIEDCOMBINETRANSITIONS($\mathcal{A}$,c,K)
  $\mathcal{A} = \langle w_g, \mathcal{N}_g \rangle_{g=1,\ldots,n}$ is a hybrid mixture of n Gaussians representing the transitions in $\tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t)}, \mathbf{Q}^{(t+1)} | \mathbf{O}_{\Delta}^{(t)} = \mathbf{o}_{\Delta}^{(t)}, \mathbf{O}^{(t+1)} = \mathbf{o}^{(t+1)}\right)$.
  c is the threshold for the similarity metrics Ji for each subsystem i.
  K is the maximum allowed number of hypotheses per subsystem.
  Let $\mathcal{B}_{\text{bel}}^{(t+1)}$ be an empty belief network containing the RVs $Hi_{\text{bel}}^{(t+1)}$, $\Delta i_{\text{bel}}^{(t+1)}$, and $\Gamma i_{\text{bel}}^{(t+1)}$ for each subsystem i as illustrated in Figure 3.6.
  Sort $\mathcal{A}$ in order of decreasing weights w_g.
  for each subsystem i **do**
    $\mathcal{A}' \leftarrow$ marginalise out all RVs not in $\{\mathcal{R}i_{\Gamma}^{+}, \mathcal{R}i_{\Gamma}^{-}\}$ from every Gaussian, $\langle w_g, \mathcal{N}_g \rangle_{g=1,\ldots,n}$, in $\mathcal{A}$.
    Mark all Gaussians in $\mathcal{A}'$ as unused.
    Set $u = 1$.
    while $u < K$ and $\mathcal{A}'$ contains unused Gaussians **do**
      Let $\langle w, \mathcal{N} \rangle$ be the first unused Gaussian in $\mathcal{A}'$.
      Find all indices of Gaussians, $l \in L$, such that $\langle w_l, \mathcal{N}_l \rangle \in \mathcal{A}'$ is unused and $Ji(\mathcal{N}, \mathcal{N}_l) \leq c$ according to Equation 3.20.
      Mark $\langle w, \mathcal{N} \rangle$ and $\langle w_l, \mathcal{N}_l \rangle_{l \in L}$ as used.
      $w' \leftarrow w + \sum_{l \in L} w_l$.
      $\Pr\left(Hi_{\text{bel}}^{(t+1)} = hi_{\text{bel},u}^{(t+1)}\right) \leftarrow w'$.
      Divide w and all w_l by w', i.e. normalise.
      Let $\mathcal{N}''$ be the Gaussian resulting from collapsing $\langle w, \mathcal{N} \rangle$ and $\langle w_l, \mathcal{N}_l \rangle_{l \in L}$ according to Equations 2.36 and 2.37.
      $\Pr\left(\Gamma i_{\text{bel}}^{(t+1)} | Hi_{\text{bel}}^{(t+1)} = hi_{\text{bel},u}^{(t+1)}\right) \leftarrow$ condition $\mathcal{N}''$ on $\mathcal{R}i_{\Gamma}^{-}$.
      for each entry r in $\Pr\left(\mathcal{R}i_{\Delta}^{(t)}, \mathcal{R}i_{\Delta}^{(t+1)}\right)$ **do**
        $r \leftarrow 0$.
        Find all transitions among $\langle w, \mathcal{N} \rangle$ and $\langle w_l, \mathcal{N}_l \rangle_{l \in L}$ whose discrete RVs assignments match those for r and add their weights to r.
      end for
      $\Pr\left(\Delta i_{\text{bel}}^{(t+1)} | Hi_{\text{bel}}^{(t+1)} = hi_{\text{bel},u}^{(t+1)}, \mathcal{R}i_{\Delta}^{-(t+1)}\right) \leftarrow$ marginalise out $\mathcal{R}i_{\Delta}^{(t)}$ from $\Pr\left(\mathcal{R}i_{\Delta}^{(t)}, \mathcal{R}i_{\Delta}^{(t+1)}\right)$ and condition on $\mathcal{R}i_{\Delta}^{-}$.
      Increment u.
    end while
  end for
  return $\mathcal{B}_{\text{bel}}^{(t+1)}$.
end procedure

3.4 Process modelling using DBNs

At this point all core concepts describing the inference engine for our PD approach have been presented, but little attention has been given to the DBN models upon which the inference engine will operate. In our case, the purpose of a DBN model is to approximately represent process dynamics — since exact representations are typically not possible due to process dynamics being linearised and time being discretised in these models.

The DBN model consists of two parts: an initial process state, and a transition model. The initial process state is represented by an initial BN whose structure is the same as the intra time slice (within a time slice) structure of the transition model. The nodes in the initial BN represent values of PVs and statuses of process components at time slice 0, and their CPDs describe an initial probability distribution over the process state when multiplied together. The transition model uses a 2-TBN to describe (approximately) the transition of a belief about a process state between successive time slices. The structure of the 2-TBN and its node's CPDs are arguably the most important parts of our PD approach. Since the same 2-TBN is used to describe all possible transitions of a process state, it needs to be carefully developed and specified to closely model real process dynamics.

This section begins with a discussion of the methods used in this thesis to develop a suitable structure for the 2-TBN.[15] The intra time slice structure of this 2-TBN can also be re-used to define the structure of the initial BN. Next, a specialised approach for dealing with non-linear PV relationships is presented, before presenting our methodology for determining the node CPDs in the 2-TBN and initial BN.

3.4.1 Defining the 2-TBN structure

The 2-TBN transition model can be procedurally created from a fundamental process model and existing process knowledge. This is further described in the ensuing subsections.

3.4.1.1 Generating a skeleton 2-TBN from a fundamental process model

A skeleton 2-TBN is a 2-TBN which describes (the structure of) a fundamental process model. It is a 2-TBN which describes the normal operating behaviour of a process. Lerner *et al.* (2000) describe an approach to generate a skeleton 2-TBN from a process description which involves using Bond and temporal causal graph modelling techniques to produce fundamental process models and using those models to generate a skeleton 2-TBN.

[15] Note that this section also contains new methods which extend upon existing methods presented in Lerner *et al.* (2000), Lerner *et al.* (2000), and Yu and Rashid (2013).

These techniques are not typically used in fundamental process modelling by engineers with chemical and mineral processing backgrounds – in most cases, process models are described by equations (see Section 2.2). Therefore, assuming availability of such equation-based models, it is unnecessary to follow the approach proposed by Lerner *et al.* (2000).

In fact, the same skeleton 2-TBN can be generated from process model equations by exploiting the particular structure of these equations. This is described in Algorithm 3.

Algorithm 3 Generate a 2-TBN from DAEs describing a process model.

Let Φ be a set of DAEs describing a process model and also let $\mathbf{X}$ be the set of variables in Φ, not including any constants or model parameters. Furthermore, the DEs in Φ must be in the standard form, i.e. $\frac{\mathrm{d}X_i}{\mathrm{d}t} = f(\mathbf{X})$, and no AE may define any X_i already defined by a DE. Let $\mathcal{B}_{\rightarrow}$ be the 2-TBN of the DBN model to be constructed.

1. For each variable in $\mathbf{X}$ create a node in $\mathcal{B}_{\rightarrow}$ at time slice t and $t+1$
2. For each AE in Φ, create an edge in $\mathcal{B}_{\rightarrow}$ from the variables on the right hand side of the equation to the variable on its left hand side in both time slices. Note that this implies assumed causal direction in the AEs even though edges in DBNs are not necessarily causal.
3. For each DE in Φ, create an edge in $\mathcal{B}_{\rightarrow}$ from the variables on the right hand side of the equation to the variable on its left hand side in both time slices. Also add an edge across the two time slices between the of the variable on the left hand side of the equation.

Appendix B.3.1 demonstrates the use of this method for generating a skeleton 2-TBN from a fundamental model of a two-tank system. The structure of the skeleton 2-TBN can then be refined to include sensor models, fault scenarios, control systems and so forth. Alternatively, one may alter the time granularity of the 2-TBN to model process dynamics, sensors, faults, and control systems differently. This is further discussed in the next subsection before moving on to the various modifications that can be made to a skeleton 2-TBN.

3.4.1.2 Choosing DBN time granularity

A DBN models process dependencies at a specific time granularity, which may differ from its associated mathematical process model. The appropriate time granularity for a DBN model depends on the sampling rate of the sensors as well as the dynamics of the process being modelled. Assuming all sensors sample at the same rate, there exists a trade-off between choosing to model the DBN using a small time granularity or large time granularity – this refers to the amount of real-time which passes between

successive time slices. A small time granularity allows the DBN to capture fast process dynamics at the cost of its time slices not synchronising with new sensor measurement schedules. This means that multiple time slices may need to be processed between new observations and could be wasteful. On the other hand, choosing to model the DBN using a larger time granularity, allows time slices of the DBN to be synchronised with new sensor measurement schedules. When performing inference for a time slice in such a DBN, one can thus expect new sensor measurements to be available for that time slice.

Lerner (2002) discusses this problem in the context of a reverse water gas shift process which features a variety of different process dynamics, from near instantaneous to the order of hours, and sensors that sample on a per-second basis. Ultimately Lerner (2002) remarks that there is little to gain from modelling a DBN using a time granularity smaller than the sensor sampling period, due to the lack of new sensor data being incorporated into the beliefs in each time slice.

Therefore, in this thesis, the time granularities of all DBNs are chosen to be the same as those of the sensor sampling periods. We further approximate any process dynamics that are faster than the time granularity of the DBN as intra time slice by modelling the quasi-steady-state behaviour of the process. To illustrate, consider a controller's manipulation of a tank's outflow: this is modelled as occurring within a time slice, whereas the effect thereof (i.e. of the change in that outflow) on the contents of that tank is modelled as occurring across time slices.

3.4.1.3 Modelling sensors

Sensors provide an interface for observing actual process behaviour. If a skeleton 2-TBN describes actual process behaviour, then sensor models added to that 2-TBN provide an mechanism for incorporating observations. In other words, sensor models describe probability distributions for observations of PV values, whereas the nodes in the skeleton 2-TBN describe probability distributions for the underlying (actual) PV values. This means that for each observed PV, a DBN contains two nodes: one for the actual value and another for the observation.

Sensors are typically noisy and add white noise to actual PV values. Therefore, a node describing an observed PV value includes an additive sensor noise component which increases the variance of the probability distribution for the observed PV value given the actual PV value. Furthermore, when a sensor is functioning normally, the mean of that probability distribution should be similar to the actual PV value.

On the other hand, when a sensor is faulty the probability distribution for the observed PV value may behave entirely differently. One typical case is that of a biased sensor. In this case, the mean of the probability distribution for an observed PV value may be quite different to the actual PV value, while the variance thereof (i.e. for the

observed PV) remains the same as the sensor's normal variance (i.e. that defined by the white noise). In other cases, the sensor may report some uniformly random value, zero, or no reading at all. However, this thesis only considers the cases of normally functioning and biased sensors.[16]

Figure 3.7 shows how we model a sensor in a DBN (Lerner, 2002). S describes the status of the sensor (i.e. *normal* or *faulty*). If the sensor is functioning normally, then its reading at time slice $t+1$ is the underlying PV value ($\mathrm{PV}^{(t+1)}$) plus the sensor's white noise component. Alternatively, if the sensor is faulty, then its reading at time $t+1$ is the sum of the sensor's bias ($B^{(t+1)}$), $\mathrm{PV}^{(t+1)}$, and the sensor's white noise component.

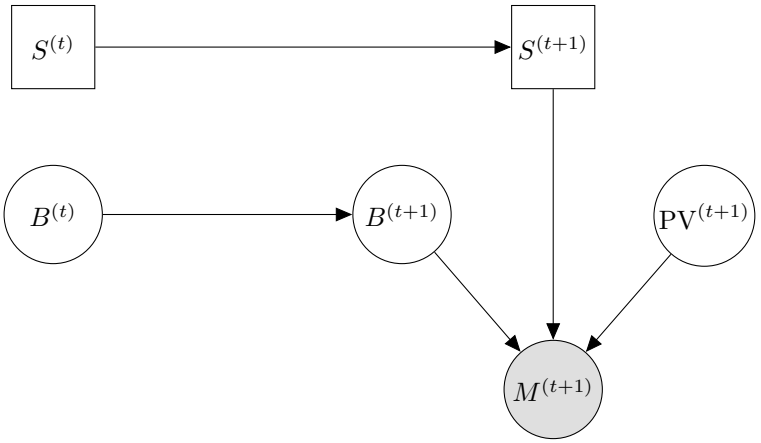


Figure 3.7: Illustration of modelling a sensor model in a DBN. S represents the sensor's status, M the sensor measurement, PV the underlying PV value, and B the sensor's bias.

The bias term RV:

Modelling a sensor in the above manner allows one to estimate the sensor's bias from observations of process behaviour. It also allows for the sensor's bias term B to change over time, enabling it to grow in situations where the sensor drifts out of calibration.

Biases are initialised with a mean of zero and reasonably large variance – in this thesis, the square of underlying PV's nominal value). Growth of the bias over time can be modelled using a linear Gaussian, $B^{(t+1)} = B^{(t)} + V$, where V is a white noise component with relatively small variance – in this thesis, a small fraction 0.01% of the initial variance assigned to the bias).

To ensure that B does not grow spuriously over time, Lerner (2002) also introduced a contraction factor, ψ, to the model for $B^{(t+1)}$ – in this case $B^{(t+1)} = \psi \cdot B^{(t)} + V$,

[16] Different sensor models may be investigated in future work.

where ψ has a range between 0 and 1. In this thesis, the value of ψ is chosen to be the same as in Lerner (2002), viz. 0.9999.

Faulty sensors:
New observations processed by the inference engine influence the belief of B over time, enabling better estimation of sensor bias over time. However, in our case it is more difficult to distinguish between a normally functioning and a faulty sensor given an observation while the sensor's bias term is near zero. This happens because the normally functioning sensor's probability distribution over the observed RVs has the same mean as that for the faulty sensor. Note that these situations were not discussed in Lerner (2002), therefore it is unclear how this was dealt with in the original work.

In this thesis, it is assumed that sensor variance for a normally functioning sensor is known. When a sensor functions normally, the observed value probability distribution should typically be highly peaked around the actual underlying PV value. On the contrary, when a sensor is biased or faulty, it is expected that this distribution should be less peaked. The sensor model used in this thesis, therefore, chooses one of two different white noise variance levels to add to the underlying PV value.[17] These white noise variance components depend on the status of the sensor and are denoted as $\sigma^2_{0,n}$ for normal and $\sigma^2_{0,f}$ for faulty status.

Since sensor variance for a normally functioning sensor is assumed to be known, $\sigma^2_{0,n}$ is known. It is also possible adjust $\sigma^2_{0,f}$ in relation to $\sigma^2_{0,n}$. In particular, when given the same mean, the Gaussian with variance $\sigma^2_{0,f}$ should be less peaked than the Gaussian with variance $\sigma^2_{0,n}$. This means that the PDF for the Gaussian with variance $\sigma^2_{0,f}$ will (for the first time) intersect that for the Gaussian with variance $\sigma^2_{0,n}$ some (real) number of standard deviations away from the mean value (as illustrated in Figure 3.8). Therefore, the desired relation between $\sigma^2_{0,f}$ and $\sigma^2_{0,n}$ can be specified by choosing a number of standard deviations away from a given mean at which point the two PDFs should first intersect.

To illustrate this, let $\sigma^2_{0,n} = 1$, the given mean equal 0, and the first intersection point for the PDFs be 2 standard deviations away from the mean. This is shown in Figure 3.8.

[17] Note that this modelling decision is not consistent with actual sensor behaviour, but is used intentionally.

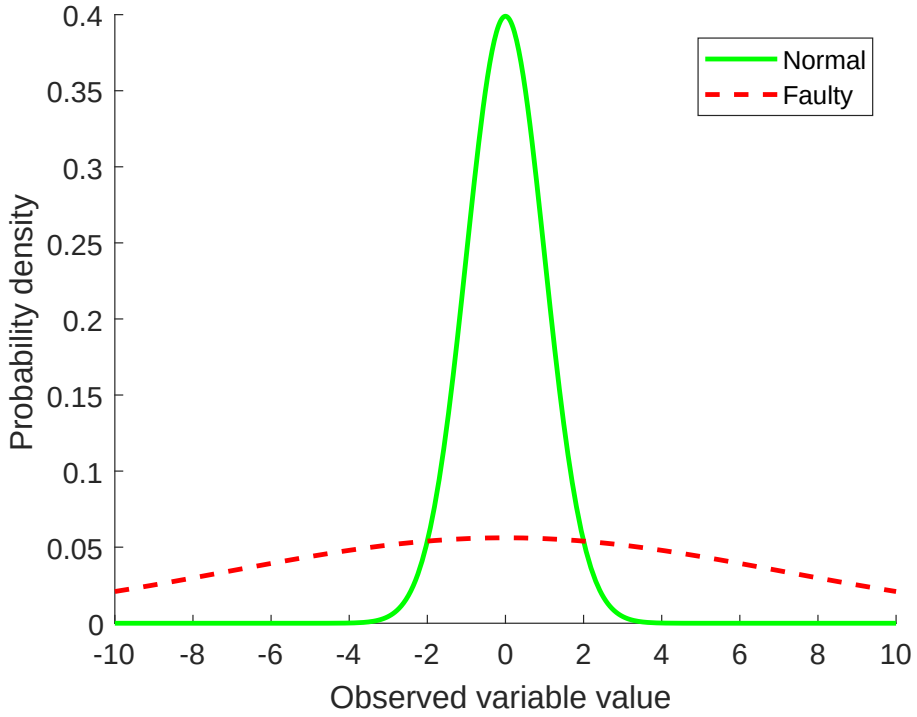


Figure 3.8: Illustration of a sensor model's observed value probability distributions. The variance for a faulty sensor is modelled such that the PDFs first intersect two standard deviations away from the same mean.

In this thesis, the given mean value used to determine $\sigma^2_{0,f}$ for each sensor is the nominal value of the actual underlying PV incorporated by linear Gaussian. The PDFs are all chosen to intersect three standard deviations away from these values for each sensor.

It is important to note that $\sigma^2_{0,n}$ and $\sigma^2_{0,f}$ are only base variances for the linear Gaussian model which describes an observed PV value. Mathematically, the full CPD structure for a sensor is shown in Table 3.4.

Table 3.4: CPD at time slice $t+1$ for a sensor. S represents the sensor's status, M the sensor measurement, PV the underlying PV value, and B the sensor bias. The sensor's white noise component is $\sigma^2_{0,n}$ when normal and $\sigma^2_{0,f}$ when faulty.

S	$\Pr\left(M \mid S, PV, B\right)$
normal	$\mathcal{N}\left(\mu_M = \mu_{PV}, \sigma^2_M = \sigma^2_{0,n} + \sigma^2_{PV}\right)$
faulty	$\mathcal{N}\left(\mu_M = \mu_{PV} + \mu_B, \sigma^2_M = \sigma^2_{0,f} + \sigma^2_{PV} + \sigma^2_B\right)$

3.4.1.4 Modelling faults as changes in process model parameters

Parameter change faults in a DBN are modelled in a similar manner to sensor biases, i.e. enabling model parameter values to drift over time under the fault condition (Lerner, 2002). The key difference is that instead of the component status influencing the probability distribution of a measurement, it influences that of a PV.

Consider a parameter change fault such as a change in the constriction resistance of a pipe in the two-tank system in Appendix B. A portion of a DBN modelling such a fault is shown in Figure 3.9.

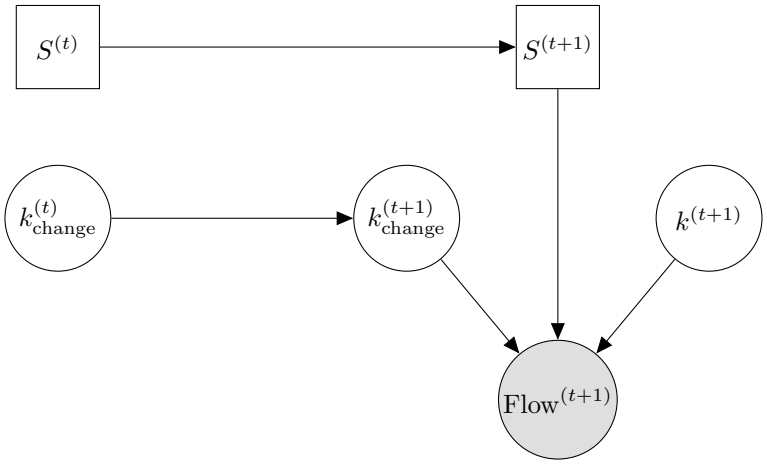


Figure 3.9: Illustration of a model parameter fault model in a DBN. S is the model parameter's status, k is the model parameter, k_{change} is a bias term for the model parameter, Flow is a PV affected by changes in k.

Note here that the status RV, S, describes the status of the flow RV. A *normal* status describes the CPD of the flow RV under nominal operation, in this case the flow would be dependent only on the constriction resistance, k, and the tank pressure, P, as in Equation 2.15 and would have reasonably low variance. On the other hand, a *faulty* status changes the dependency of the flow RV as shown in Equation 3.21

$$F = \left(k + k_{\text{change}}\right)\sqrt{P} \tag{3.21}$$

The RVs k_{change} is a bias term for k and has the same CPD as that of a sensor bias, allowing it to grow slowly over time but tending toward zero as result of a contraction factor. Additionally, this means that when the status for Flow is faulty, the underlying PV value's variance must be higher than that when the status for Flow is normal to avoid spurious detection of fault conditions. In this thesis, the same approach as for faulty sensor variances is used here. However, the increased variance for a faulty status is instead determined in relation to the variance (under NOC) of the underlying PV values.

3.4.1.5 Dealing with regulatory control and recycle structures

Regulatory control and recycle structures in processes pose an interesting problem for modelling in a DBN because these structures are inherently cyclic and BNs are required to be acyclic. Yu and Rashid (2013) deal with this problem by adding additional "dummy" RVs to each time slice of a DBN. These RVs account for the cyclic interaction between the PVs involved. For example, consider the interaction between a manipulated (MV) and controlled (CV) variable in a control loop; the MV inherently has an open-loop influence on the CV, but the CV influences the MV as result of feedback control. Yu and Rashid (2013) model this as shown in Figure 3.10 for one time slice. It should be noted that this approach assumes that the dynamics of the control loop is faster than the time granularity of the DBN.

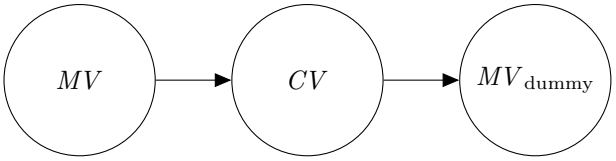


Figure 3.10: Illustration of the BN modelling approach for a regulatory control structure used in Yu and Rashid (2013). MV is the manipulated variable, CV is the controlled variable, and MV_{dummy} is an additional variable used to model the feedback effect.

Dealing with regulatory control and recycle structures in this manner, however, is not sufficient for designing DBNs for use with our fault diagnosis approach. This is because the DBNs used in the approach by Yu and Rashid (2013) only model NOC , while this work also considers a suite of possible faults. Therefore, control and recycle structures need to be modelled more explicitly in a DBN for use in our approach.

Regulatory control structures

Marlin (2000) presents two approaches for digital implementation of the proportional-integral-derivative (PID) control algorithm, namely the full-position and velocity implementations. These implementations are also useful for modelling proportional-integral (PI) control loops in a DBN because time is discrete for the controller implementation and the DBN. In this thesis, the full-position implementation is used since it incorporates the initial controller output. We discuss it next.

This implementation assumes that the controller output, MV, remains constant between time slices and updates as new sensor readings are observed. The controller calculations are similar to that of an analogue controller, in that the time t controller output is based on the difference between the controlled variable's set point value and its observed value at time t, namely the error E_t:

$$E_t \equiv SP_t - CV_t. \tag{3.22}$$

The full-position digital controller implementation approximates the integral mode of the controller by a Riemann sum. This is done by accumulating the error from previous executions of the controller in $S_{\text{acc},t}$ and adding to it the current error, E_t, as shown in Equation 3.23.

$$\begin{aligned} MV_t &= K_c \left(E_t + \frac{\Delta t}{T_I} S_{\text{acc},t} \right) + MV_{\text{nominal}} \\ S_{\text{acc},t} &= \sum_{i=1}^{t} E_i = E_t + S_{\text{acc},t-1} \end{aligned} \tag{3.23}$$

where $S_{\text{acc},t-1} = \sum_{i=1}^{t-1} E_i$ is stored from the previous controller execution. Furthermore, K_c and T_I are controller parameters, Δt is the time granularity of the DBN, and MV_{nominal} is the initial controller output.

Figure 3.11 shows the full-position PI controller implementation in a portion of a DBN.

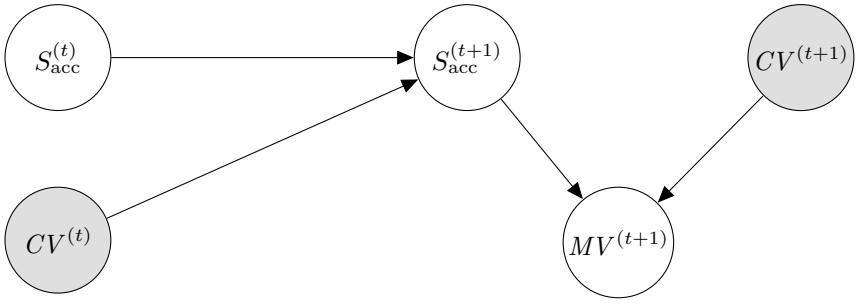


Figure 3.11: Full-position PI controller implementation in a portion of a DBN.

Recycle structures

Recycle structures are simpler to model in a DBN than regulatory control structures because these structures do not explicitly require storing or accumulating previous values of variables. Furthermore, we limit our attention to flow stream recycle, it is fair to assume that the dynamics of the recycle is faster than the time granularity of the DBN. Therefore, the flow through the recycle stream is determined (almost) instantaneously within a time slice and propagates across time slices to affect PVs in the next time slice, i.e. the effects of the recycle stream is modelled as an inter-time slice edge in a DBN. Note that in systems with slower recycle dynamics, this may be a bigger issue.

3.4.2 Dealing with non-linear PV relationships

An important practical issue is the modelling of non-linear relationships between the continuous RVs in a DBN model of a process. Recalling Section 2.3.2, PVs are often modelled as Gaussian RVs with linear relationships between them. Typically, these

linear Gaussian relationships represent linearised forms of the non-linear relationships that exist between these PVs. Therefore, linearisation yields β_0 and $\boldsymbol{\beta}$ for each linear Gaussian relationship between PVs.

Although these linear representations of non-linear functions are not ideal, the family of Gaussian distributions is more computationally convenient and not using this may (and typically does) make inference more complex and typically intractable. Therefore, remaining within the Gaussian family of continuous probability distributions is more desirable than attempting to model non-linear functions exactly. The unscented transform (Julier and Uhlmann, 2004) enables just-in-time linearisation of non-linear functions and is typically a superior linearisation method when compared with Taylor series approximation (Marlin, 2000). The linearisation approach used in this thesis is further detailed below.

Given a non-linear function, $Y=f(\mathbf{x})$, of the PVs, $\mathbf{X}$, and the probability distribution, $\Pr(\mathbf{X})$, over $\mathbf{X}$; f may be linearised by sampling from $\Pr(\mathbf{X})$, propagating the samples through f, and finding β_0 and $\boldsymbol{\beta}$ by means of regression (i.e. from the propagated function values to the samples). Let $\mathbf{S}$ be a collection of samples such that each sample, $\mathbf{x}_n$[18] in $\mathbf{S}$, when propagated through f yields a new $y_n = f(\mathbf{x}_n)$. The idea is to generate a few y_n which may be used to approximate the probability distribution over Y by a Gaussian with mean μ_y and variance σ_y^2. Assuming that $\mathbf{X}$ has the distribution $\mathcal{N}(X : \boldsymbol{\mu}, \boldsymbol{\Sigma})$, it is desired to determine the $\mathbf{S}$ that maximises the information (Cover and Thomas, 2005) gained from each y_n for the approximation of $\Pr(Y)$. The unscented transform is a technique which does exactly this and approximates $\Pr(Y)$. The technique is well established and is part of the freely available EKFUKF toolbox for MATLAB® by Hartikainen *et al.* (2011).

Although the unscented transform offers direct approximation of $\Pr(Y)$, limitations in practice[19] means that this cannot be used directly and needs to be integrated with the linear Gaussian framework. Thus we need to compute the weights β_0 and $\boldsymbol{\beta}$ for $\Pr(Y|\mathbf{X})$ by way of regression using results obtained from the unscented transform.

Furthermore, since f may be any function, the use of the unscented transform also allows the application of constraints to the output of f for each $\mathbf{x}_n$. For example, let Y be the flow, F, and $\mathbf{X}$ be the pressure, P, in the Bernoulli Equation 2.15 and suppose an $\mathbf{x}_n(P)$ is negative (and thus not physically possible), f may be defined as piece-wise dependent on the value of $\mathbf{x}_n(P)$ such that it outputs 0 when $\mathbf{x}_n < 0$ and Equation 2.15 elsewhere. This may limit the extent to which $\Pr(Y)$, obtained from the unscented transform, exceeds the physical limitations of some PV values.

[18] Where each entry in $\mathbf{x}_n$ is an assignment to one RVs in $\mathbf{X}$.

[19] The Bayesian Network Toolbox (BNT) (Murphy, 2002) currently does not support non-linear Gaussian CPDs.

This linearisation method was tested on the draining liquid level case study in Appendix A.3. The results thereof showed minimal loss in process state tracking ability when all observations are close to actual process behaviour and linear degradation in process state tracking ability when all observations are completely inaccurate.

3.4.3 Defining CPD parameters in DBNs

There are two sets of CPDs that are used to define a DBN model for a process. The first set is the initial RVs CPDs, i.e. representing time slice 0. These CPDs are relatively easy to define and may be determined from component reliability data for the discrete RVs and from historical process data for the continuous RVs. The second set is the 2-TBN CPDs which collectively define all possible process state transitions between successive time slices. These are more difficult to define and may require some trial and error tuning to achieve desired diagnostic performance.

In this section, we present the methods used to define CPDs for the DBN models used in this thesis. The methods presented here aim to provide some suggested rules-of-thumb and guidelines for determining CPD parameters, these may be helpful for future work using this PD approach.

Initial process state CPDs These consist of both discrete and continuous RVs CPDs. The discrete RVs CPDs are discussed first. These are marginal probability distributions describing the initial status for each component modelled in the DBN. In practice, the probability of a component being in a faulty (or defective) status may be determined by the reliability of that component. However, in this thesis all components are assumed to have equal reliability and thus a single initial probability of fault value is used to define all discrete CPDs in the initial process state. This value is discussed in further detail when the component status CPDs are considered, since it is chosen to be the same as the probability for the transition of a component in the normal status to a fault status. Note that since we use a single initial probability of fault value, we also choose the initial probability of each fault status for a component as being equal to this value.

The continuous RVs CPDs are either Gaussian or linear Gaussian. For the Gaussian CPDs, the mean and variance of a PV may be estimated from historical process data. This is done by fitting a Gaussian distribution to the observed values of each PV over time, specifically under NOC . Note that although these observed RV probability distributions (Gaussians) include the additive white noise variance due to sensor noise, this is not subtracted when estimating the variance for the CPDs in the initial process state. This is because subtracting sensor noise at this point would imply high certainty about distribution of PVs in the initial state, which may result in incorrect tracking of the state over time. On the contrary, being less certain about the distribution of PVs in the initial state allows for inferred probability distributions of PV to become better over time, i.e. with respect to higher certainty about the process state. This is further

discussed when we consider the PVs CPDs which are not part of the initial process state.

In this thesis, most linear Gaussian CPDs will have their β_0 and $\boldsymbol{\beta}$ determined by the just-in-time linearisation method described in Section 3.4.2. Allowing a set of non-linear functions to be specified as a property for each continuous RV node in a DBN facilitates the use of this method. Next, given β_0, $\boldsymbol{\beta}$, and the means of the CPD's node's parents, the mean of the marginal of these parents can be computed as per Equation 2.40. For this reason it is not necessary to specify a mean value parameter for such a linear Gaussian CPD[20]. On the other hand, computation of the variance of the marginal using the same equation, requires further specification of a base variance parameter, i.e. σ_0^2. This base variance is determined in the same way as previously described for Gaussian CPDs.

Further note that the continuous RVs CPDs for the initial process state are the same regardless of process operating mode and are duplicated for the cases of abnormal operating modes.

Component status CPDs These CPDs describe the transition of discrete RVs from time slice t to $t+1$, i.e. by defining the probability that a component transitions from a normal to a normal/abnormal status or from an abnormal to a normal/abnormal status. The (approximate) process state at time slice t describes a probability distribution for each component's status at time slice t. Recall that these probability distributions are the result of combining various hypotheses, which attempt to explain observations for time slice t, with one another. Also note that the component status CPDs affect the likelihoods of computed transitions which, in turn, affects the likelihoods of the hypotheses created from those transitions.

It is assumed that process components generally have low likelihood of showing abnormal behaviour. Because of this, the probability for a component transitioning from a normal to an abnormal status should be low. This prevents spurious detection of faults in components which had been previously determined to be functioning normally with high likelihood. In this thesis, a value, $\alpha_{\text{nf}} = 0.001$ is assumed as the probability for a component transitioning from a normal to an abnormal status over a single time step. However, note that for processes with many operating modes (due to the number of components or the number of component statuses) this value is too large.[21] Therefore, a Bonferroni correction (Miller, 1981) is applied to scale this value with respect to the number of operating modes, m_{opmodes}, the DBN of the process can encode. The number of operating modes is the product of the number of statuses for

[20] Note that the BNT (Murphy, 2002) requires the user to specify a mean value parameter for linear Gaussian CPDs, for the aforementioned reasons this is set to zero in our implementation.

[21] This is due to the problem of multiple comparisons (Miller, 1981) – an increased number of operating modes results in an increased number of possible hypotheses which describe abnormal process behaviour all of which are tested for significance (or likelihood in our case) simultaneously.

all the components in a DBN — in this thesis all components have only two possible statuses. A Bonferroni corrected probability for a component transitioning from a normal to an abnormal status is calculated by dividing α_{nf} by m_{opmodes}. This value is also used to define the time slice 0 probability for a fault status in a component. Together with low transition probability from a normal to an abnormal status, this makes it more difficult for fault conditions to be spuriously reported.

A uniform probability distribution is assumed for fault status transitions, i.e. fault-to-fault and fault-to-normal. Doing this ensures that spurious hypotheses which report fault conditions do not perpetually grow in likelihood over time when observed process behaviour does not agree with them.[22]

Table 3.5 illustrates the parameters for a component status CPD for a component which has two possible statuses, i.e. normal and faulty.

Table 3.5: Modelling of 2-TBN transition probabilities from time t to $t+1$ for a component with two statuses.

$S^{(t)}$	$S^{(t+1)}$	$\Pr\left(S^{(t+1)}\middle\vert S^{(t)}\right)$
normal	*normal*	$1-\frac{\alpha_{\text{nf}}}{m_{\text{opmodes}}}$
normal	*faulty*	$\frac{\alpha_{\text{nf}}}{m_{\text{opmodes}}}$
faulty	*normal*	$\frac{1}{2}$
faulty	*faulty*	$\frac{1}{2}$

PV CPDs These linear Gaussian CPDs describe the intra time slice relationships between PVs in a DBN, i.e. they describe process behaviour using the time scale of the DBN model. This procedure uses uncertain inputs, i.e. the probability distributions of PVs at time slice t, to determine probability distributions of PVs at time slice $t+1$. Additionally, non-linear relationships between PVs are linearised here using the unscented transform from Section 3.4.2. This means that uncertainty about distributions of PVs at time slice $t+1$ comes from uncertainty about those values at time slice t, error introduced during linearisation, and modelling inaccuracies. Note that although the linear Gaussian framework compensates for uncertainty in PVs at time slice t as per Equation 2.40, it does not consider uncertainty in the values of β_0 and $\boldsymbol{\beta}$.

Therefore, the σ_0^2 parameter for each PV CPD is used to define a base variance for that PV. This base variance can be estimated from historical process data for each PV given the variance of the sensor.[23] Since sensor noise is assumed to be additive

[22] Note that this approach does not, nor attempts to, realistically model fault status transitions for process components.

[23] Note that all PVs are monitored by sensors in this thesis. Therefore in practice, these base variances may need to be determined from functional relationships between monitored and unmonitored PVs.

white noise, the sensor noise simply needs to be subtracted from the variance of the observed PVs.

Note that this modelling approach, in addition to specifying higher initial uncertainty for the distribution of PVs, facilitates robust estimation for the (distributions of) values of PVs over time assuming normal operation. This is because initial uncertainty is reduced over time as more data is observed to fit the model predictions. Subsequently, the observation of anomalous data which does not match the expected distributions of PVs is more likely to cause the approach to report the detection of fault conditions.

Recall from Section 3.4.1.3 and Section 3.4.1.4 that the bias RV approach used in this thesis may result in spurious detection of fault conditions. Therefore, for a particular PV, the aforementioned base variance parameter only applies under NOC, and a different base variance parameter applies under the condition of abnormal behaviour for that PV.[24] The same approach and standard deviation threshold as for faulty sensors is used here, i.e. that the PDFs intersect three standard deviations away from the nominal value of the PV.

3.5 Conclusion

In closing, this chapter details first how the PD approach presented in Lerner *et al.* (2000) and Lerner (2002) works, and then how specific design decisions were made for this work. Our approach is similar to that of Lerner (2002) and deviates in parts where the source material was too vague. We also make use of alternative methods to what was implemented in Lerner (2002) and extend upon their approach to include cyclic process structures such as regulatory control and recycle.

It is also important to note that while we provide a reconstruction of the PD approach presented in Lerner *et al.* (2000) and Lerner (2002), our approach still makes use of simple probabilistic methods. We also implement the same computational enhancements and experienced similar implementation issues — discussed in Section 5.3.3. While it is possible to extend upon this implementation with further optimisation and simple probabilistic methods, more advanced methods may address some of our largest challenges and issues. In particular, variational inference methods such as expectation propagation (Minka, 2001; Koller and Friedman, 2009) may greatly reduce computational requirements and mitigate the need for ad hoc optimisation methods.

The next chapter contextualises our study by considering various other studies using DBN-based PD approaches in the context of chemical and mineral process fault diagnosis.

[24] Note that this only applies to PVs which are modelled to have abnormal statuses, such as in sensor models.

Chapter 4

Literature Review

Fault diagnosis in industrial operations has been applied and continuously improved for many years (Venkatasubramanian *et al.*, 2003*b*; Aldrich and Auret, 2013). Before computers became readily accessible and available to industry, fault diagnosis approaches would rely primarily on humans for fault detection and root cause analysis. This resulted in late detection of faults and/or incorrect root cause diagnosis which could escalate fault conditions on-site, resulting in significant loss. The techniques used were based on available process knowledge at the time, and produced tools such as hazard and operability (HAZOP) manuals that could be used to diagnose faults (Lees, 2012).

Early software for fault diagnosis used qualitative model-based techniques, such as expert systems (Rich and Venkatasubramanian, 1987), to reduce the effort required by human operators to isolate faults. However, human operators still had to continuously monitor the process and sensor data in order to detect faults. The first quantitative model-based fault detection techniques made use of observers (Willsky, 1976; Isermann, 1984; Chen and Patton, 1999). Typically, an observer maintains and updates an internal plant model as it observes new sensor data. Using this model, it then attempts to predict future process behaviour. Deviation of the process from the observer's predictions generates residuals which are continuously monitored, making fault detection easier. However, accurate root cause analysis while preserving robustness to uncertainty still remained elusive (Patton and Chen, 1997).

DBN-based PD approaches enable fault detection and root cause analysis in a single framework. This chapter presents the recent advancements in these approaches, focusing on how the techniques have improved with each new study. It will become apparent that approaches can not be directly compared to each other, because of the use of different testing methodologies and lacking quantitative performance metrics. Therefore, the approaches are qualitatively evaluated and compared in terms of desirable characteristics of fault diagnosis approaches (see Section 2.2). The chapter concludes with a critical analysis of the state of DBN-based PD research.

4.1 DBN-based probabilistic fault diagnosis approaches

Lerner *et al.* (2000) proposed using a DBN to unify fault detection and root cause analysis in one framework. They reason that this approach inherently models uncertainty in operations which should mitigate the lack of sensitivity in traditional qualitative model-based techniques. Their approach incorporates process knowledge directly into the structure and CPDs of a DBN with explicit models for all faults of interest.

The DBN used represents a collection of linear[1] fundamental process models (i.e. a switching linear dynamical system (Gertler, 1988)) each of which may be selected by different particular assignments to the network's discrete RVs (nodes). Every combination of values assumed by the set of discrete nodes in the network represents a single operating mode for the process, which can range from normal operation to one particular sensor fault to multiple process component faults. In this way, Lerner *et al.* (2000) reduced the task of fault detection and root cause analysis to performing inference in a DBN.

Unfortunately the DBN resulting from the approach in Lerner *et al.* (2000), is intractable for exact inference any complex process. Lerner *et al.* (2000) therefore further describe an approximate inference algorithm to track the state of the process by considering only the most probable changes in operating mode over time, i.e. the trajectories of the process state with respect to time. A first order Markov assumption is applied to avoid exponential blow-up of the number of trajectories over time. Since the set of most probable trajectories may feature multiple similar trajectories, these are combined to create a set of *hypotheses* which are used to approximate the process belief state. The approximate belief state is then used (iteratively) to detect and diagnose fault conditions in discrete time.

One major constraint of this approach is that the computational time required to process a sample increases as more operating modes are added to the DBN model. This makes fault detection slower since the number of possible trajectories are exponentially larger than the number of operational modes. Roychoudhury *et al.* (2006) tackled this problem by considering fault detection and root cause analysis in separate stages within one framework. Their approach was divided into three stages: (1) fault detection, (2) qualitative fault isolation, and (3) quantitative fault isolation.

In the fault detection stage, Roychoudhury *et al.* (2006) use a DBN, modelled under normal operation, to generate residuals for fault detection. Further analysis of the residuals over time yields qualitative deviations (i.e. increase, decrease, no change)

[1] Non-linear dynamics are typically approximated by first order Taylor series approximation in most research.

in the measured (observed) PVs which are used in the second stage. The second stage compares the patterns in the qualitative deviations to a database of fault signatures (i.e. the effects particular faults are expected to have on the qualitative deviations of the observed PVs). The fault signatures which closely match the qualitative deviations over a number of time steps are used to construct and initialise a new DBN containing models of these faults. Finally, in the third stage, a similar approach to Lerner *et al.* (2000) is used to isolate the fault cause from the reduced fault set.

It should be noted that the approach of Roychoudhury *et al.* (2006) only diagnoses incipient faults. Roychoudhury *et al.* (2008) extended this work to include the diagnosis of abrupt faults as well. The overall approach appears to be very similar in the two papers, with the latter study extending the qualitative fault isolation stage by considering fault signatures for both abrupt and incipient faults. Roychoudhury *et al.* (2008) additionally describe the use of particle filters (Koller and Lerner, 2001) for approximate inference. Results from Roychoudhury *et al.* (2008) were positive for fault detection, however, there was a fault for which the root cause could not to be isolated. The authors suggested this problem may be the result of sparse measurement information which affects the observability of the DBN.

Furthermore, the authors suggest that computational efficiency could be further improved by distributing the diagnosis task amongst reduced DBNs derived from the global DBN. Roychoudhury *et al.* (2009) showed that the same approach in Roychoudhury *et al.* (2008) could be applied to such reduced (factored) DBNs, improving computational efficiency without sacrificing the accuracy of the diagnosis. This approach applies the same techniques as in Roychoudhury *et al.* (2008) to local diagnosis modules which act on smaller subsystems. The factored DBNs of the local diagnosis modules are designed in such a way that multiple modules communicate minimal measurement information to one another, yet still achieve local diagnosis results that are globally correct.

Roychoudhury *et al.* (2010) compared the distributed diagnosis approach found in Roychoudhury *et al.* (2009) to the centralised diagnosis approach found in Roychoudhury *et al.* (2008). Roychoudhury *et al.* (2010) showed that computational efficiency was improved without sacrificing the accuracy of the diagnosis. To our knowledge, this was the last study extending upon Lerner *et al.* (2000) and Roychoudhury *et al.* (2006).

Yu and Rashid (2013) present an alternative approach to fault diagnosis using DBNs. Their approach uses existing process knowledge to obtain the structure of a DBN under NOC, including modelling of control loops and recycle streams which had not been considered in previous studies. However, their DBN only contains observed continuous PVs trained using historical process data to determine the CPDs for each node in the network under NOC, and does not model any operating modes of the process as in Lerner *et al.* (2000).

Yu and Rashid (2013) use a similar approach to Roychoudhury *et al.* (2008) for fault detection, i.e. by computing the likelihood for abnormal operating conditions based on deviation of their model predictions from observed process behaviour. However, for root cause analysis, Yu and Rashid (2013) present a substantially different approach which was inspired by the contribution plots typically used in MSPM approaches: Yu and Rashid (2013) decompose the abnormality likelihood for the entire process into abnormality likelihoods for each RV in their DBN and attempt to identify possible fault propagation pathways leading to the identified fault symptoms by exploiting the structure of the DBN — assuming that the directed edges imply causality.

To our knowledge, PD approaches which use a DBN model to directly incorporate existing process knowledge is limited to this set of studies. However, it should be noted that other researchers have attempted more naïve approaches using BNs in purely data-driven environments with some degree of success (Dey and Stori, 2005; Verron *et al.*, 2010; Zhao *et al.*, 2013; Zhang and Dong, 2014; Atoui *et al.*, 2015; He *et al.*, 2016; Askarian *et al.*, 2016; Verbert *et al.*, 2017). A major limitation of these approaches is that they often rely on the availability of data containing known faults, which is not typically readily available.

4.2 Qualitative evaluation of the state of the art

An ideal fault diagnosis approach should be applicable to any process given sufficient observability (i.e. by way of placement and number of sensors) and *a priori* knowledge (i.e. fault sets and models) of the process.

However, the case studies selected for testing fault diagnosis approaches in most literature tend to be convenient, small systems which are vaguely described and not easily replicated. These include: (1) simple systems such as two tank systems (see Appendix B) which tend to be used for presenting the inner workings of novel approaches (Roychoudhury *et al.*, 2006, 2008); (2) real physical processes which showcase the applicability of an approach in real-world scenarios (Lerner, 2002; Roychoudhury *et al.*, 2009); and (3) unique, ad-hoc, simulated processes which showcase the applicability of an approach under complex scenarios (Lerner *et al.*, 2000; Roychoudhury *et al.*, 2010).

On the other hand, the Tenessee Eastman chemical process (Downs and Vogel, 1993; Ricker, 1996) is a complex simulated case study that is widely used to test fault diagnosis approaches. Although more frequently used in data driven studies (Atoui *et al.*, 2015; Askarian *et al.*, 2016), only Yu and Rashid (2013) chose to evaluate their approach using this case study.

Clearly no direct comparison can be made between the limited quantitative results of

each study because of the authors' diverse selection of case studies. Furthermore, none of the studies of interest presented quantitative performance metrics for fault detection (Section 2.2.3). As a result, only a qualitative evaluation of each study according to the desired characteristics for a fault diagnosis approach (Section 2.2) can be presented. Table 4.1 shows which requirements are claimed to be met by the approach in each study of interest given the available information about each study and its results.

Table 4.1: Comparison of characteristics of techniques presented in literature. Studies in this table are cited as: [LPKB] Lerner *et al.* (2000), [RBK1] Roychoudhury *et al.* (2006, 2008), [RBK2] Roychoudhury *et al.* (2009), [RBK3] Roychoudhury *et al.* (2010), [YR] Yu and Rashid (2013).

Characteristics	**LPKB**	**RBK1**	**RBK2**	**RBK3**	**YR**
Quick detection and diagnosis	n.d.*	n.d.*	n.d.*	✓	✓
Isolability	✓	✓	✓	✓	✓
Robustness	n.d.*	✓	✓	✓	✓
Novelty identifiability	✗	✗	✗	✗	✓
Classification error estimate	✓	✓	✓	✓	✓
Adaptability	✓	✓	✓	✓	✓
Explanation facility	n.d.*	✓	✓	✓	✓
Modelling requirements	High	High	High	High	Mid
Storage and computational requirements	High	High	Mid	Mid	Low
Multiple fault identifiability	✓	✓	✓	✓	n.d.*

*n.d. - not disclosed

Each study in the columns of Table 4.1 tested their approach on different case studies, using different testing methodologies, and reporting different performance results. To better understand and compare differences between the performance of each approach in each study, a discussion of each study's testing methodology and results are presented below. Features of the case studies used in each study are summarised in Table 4.2.

Table 4.2: Features of the case studies used to evaluate the fault diagnosis performance of the techniques presented in literature. Studies in this table are cited as: [LPKB] Lerner *et al.* (2000), [RBK1] Roychoudhury *et al.* (2006, 2008), [RBK2] Roychoudhury *et al.* (2009), [RBK3] Roychoudhury *et al.* (2010), [YR] Yu and Rashid (2013).

Feature	LPKB	RBK1	RBK2	RBK3	YR
Simulated process	✓	✓	✓	✓	✓
Number of system volumes	5	2	8	12	5
Sensor noise	✗	✓	✓	n.a.*	✓
Control system	✗	✗	✓	n.a.*	✓
Recycle streams	✗	✗	✓	n.a.*	✓
Maximal number of PVs	11	5	17	12	41
of which observed	3	3	14	12	22
Number of faults tested	6	6	1	4	2
Abrupt/incipient manifestation	Both	Both	Both	Abrupt	Abrupt

*n.a. - not applicable

[LPKB] Lerner et al. (2000) tested their approach using a simulated case study of five tanks in series, connected by pipes at their bottoms. No sensor noise, regulatory control system, or recycle streams were present in this case study. The case study contained five pressure and six flow PVs, where three of the flow PVs were observed. A total of six fault conditions were simulated; three abrupt and two incipient faults in the flow resistances as well as one abrupt sensor bias fault in two sensors. The case study presented a challenge for accurate state estimation given the sparsity of the measurement information.

The authors tested their approach on stored data for a single run containing all six fault scenarios which occur one after the other. Although generating data in this manner is not ideal, the authors claim to produce successful detection and isolation of most of the faults within two to three time steps. This success was largely attributed to the use of a technique called smoothing which relies on the ability of the approach to process multiple samples within a reasonable time frame (in order to be of practical use).

Computation times were not reported in Lerner *et al.* (2000), therefore it is unclear whether or not this approach may be used in real-time fault diagnosis and thus unclear whether or not quick detection and diagnosis was achieved. Additionally, it is unknown if the approach is robust to noise since no noise was added in the simulated data.

It should be noted that a PhD thesis (Lerner, 2002) later extended upon Lerner *et al.* (2000) and tested the new approach using both simulated and real-world versions of a reverse water-gas shift system. However, no further quantitative fault

diagnosis performance results were reported.

[RBK1] Roychoudhury et al. (2006) and Roychoudhury et al. (2008) both tested their approach using a simulated case study of two tanks connected by a pipe at their bottoms. No regulatory control system or recycle streams were present in this case study. The case study contained two pressure and three flow PVs, where all of the flow PVs were observed. A total of six fault conditions were considered; three abrupt and three incipient faults in the flow resistances. Roychoudhury *et al.* (2006) presented results for all of the incipient faults, whereas Roychoudhury *et al.* (2008) presented results for only one abrupt and one incipient fault.

The authors simulated one set of data (including normal operation) for each fault and tested their approaches off-line, adding white noise to the measurements. It was noted that Roychoudhury *et al.* (2006) seem to show that faults were introduced into the process during a transient state (i.e. not stationary nor steady-state) which may have affected the results of the study.

Roychoudhury *et al.* (2006) report detection of the incipient faults within 5 to 19 samples (seconds) and isolation within 100 to 258 samples (seconds). Roychoudhury *et al.* (2008) report detection of both faults within 2 seconds, but do not explicitly report time for fault isolation. The approaches in these two studies seem to be similar, save for the qualitative fault isolation stage (see Section 4.1), but an improvement in diagnosis performance was notable in the results of the latter study.

The qualitative fault isolation stage allows both approaches to have improved explanation facility over Lerner *et al.* (2000). However, additional effort is required to generate fault signatures which may limit the applicability of the approach compared to that of Lerner *et al.* (2000).

[RBK2] Roychoudhury et al. (2009) tested their approach using a simulated case study of a physical operation, namely the Advanced Water Recovery System of Pickering *et al.* (2001). Regulatory control and recycle streams were present in the case study. The case study contained many PVs and was said to be well-instrumented, having seventeen of its PVs observed. A total of twenty abrupt and incipient faults were considered, including incipient build-up and abrupt partial blockages in piping, decrease in equipment (pumps, blowers, heaters) efficiency and other system-specific faults.

The study aimed to test the viability of a distributed diagnosis approach using factored DBNs. Two approaches were developed, one assuming user-specified subsystems and the other generating its own subsystems. Three simulated data sets containing a reduced set of faults and varying degrees of observability (maximal, intermediate and minimal) were used to test each approach. However, results were only

reported for one fault using the first approach, i.e. based on user-specified subsystems.

The choice of fault was an abrupt partial blockage in one of the pipes, where white noise was added under minimal observability. The authors show the process of refining the set of faults, based upon their fault signatures, in each of the distributed diagnosers. The local diagnosis in each distributed diagnoser was shown to produce a globally correct diagnosis for the fault. Quantitative fault detection and isolation times were not reported.

[RBK3] Roychoudhury et al. (2010) tested their approach using a simulated case study of a twelfth order electrical system. The case study contained many PVs, ten of which were observed. A total of twenty-two abrupt and incipient faults were considered, including changes in capacitances, inductances, and resistances. However, results were only reported for four abrupt faults in the set.

Two approaches were tested, one using distributed diagnosers and the other using a centralised diagnoser. Fault detection time ranged from 0.2 to 118.3 seconds and from 0.2 to 196.8 seconds for the distributed and centralised approaches respectively. Similarly, fault isolation time ranged from 3.02 to 163.3 seconds and from 2.8 to 377.4 seconds for the distributed and centralised approaches respectively. Ultimately, the results supported the authors' claim that the distributed diagnosis approach would be computationally less expensive than its centralised counterpart, without loss in the accuracy of the diagnosis.

[YR] Yu and Rashid (2013) tested their approach on the widely-used simulated case study created by the Tenessee Eastman company (Downs and Vogel, 1993; Ricker, 1996). Noise, regulatory control, and recycle streams are present in this case study. The case study contains forty-one measurable output PVs, twenty-two of which are observed. Two abrupt faults (of a possible twenty) were considered: the first was a decrease in one of the operation's feed streams and the second was an increase in the random variation of the reactor pressure.

The simulated data contains both measurement noise as well as random process disturbances in order to mimic realistic data. The authors showed (visually) that the approach was able to produce a low false alarm rate and was able to detect each fault quickly with low misclassification rates, but do not report quantitative results. Additionally they show that their approach was able to correctly identify the root cause of both faults as well as both of their propagation pathways.

In contrast to the approaches in the previous studies, this approach does not track the evolution of the process's state over time. Therefore, it may not perform as well for diagnosis of incipient faults (not considered in the study). However, the

integration of data-driven techniques with (non-naïve) probabilistic models shows promising results for fault diagnosis in complex processes.

4.3 Remarks

4.3.1 Comparability of state of the art approaches

It was not possible to quantitatively compare the studies discussed in this chapter due to their use of different case studies, different testing methodologies, and incomplete reports of quantitative fault diagnosis performance results. It is desirable that some standardisation of case studies, testing methodology, and performance metrics be established for quantitative future comparison of fault diagnosis approaches.

One way to present case studies for better comparative evaluation may be to discuss the features of each case study. Any process may be characterised by its size (number of system volumes, number of measurable PVs), complexity (sensor noise, process disturbances, control loops, recycle streams), faults (number, type, severity, frequency), observability (number of observed PVs), model availability (theoretical/empirical), and the time scale of causal relationships between PVs (for processes with both slow and fast dynamics). Comparison of performance metrics, even for different case studies, may be more useful given better comparative information about each case study. This would enable rough extrapolation of performance to different case studies which have similar characteristics.

It is also important that simulated data from case studies are produced starting from a stationary state with noise and process disturbances present before introducing a single fault into the process for each data set. This would allow for more direct comparison of fault detection performance by means of the binary classification metrics discussed in Section 2.2.

4.3.2 Suitability of DBN-based probabilistic fault diagnosis

DBN-based PD approaches directly incorporate process knowledge into a DBN model, producing good isolability of faults as found in the studies presented in this chapter. The model-based nature of the approach generally makes it more adaptable to process changes (such as set point or minor structural changes) in comparison to a purely data-driven approach, since fault diagnosis is reduced to the task of inference in a DBN. This allows for models to be updated and revised without the need to retrain a purely data-driven approach. However, CPD parameters may still need to be updated (possibly from data).

Naturally, this approach has higher modelling requirements since process knowledge

is required to develop the structure of the DBN as well as to determine the values or parameters in the CPDs for each node (although these can also be learned from data, when available). Furthermore, computational speeds vary with the specific details of the approach, especially due to the way in which the DBN is designed and the way in which the inference engine is implemented.

Although the studies discussed in this chapter show promising results for root cause analysis, their fault detection performance was seldom quantified. Furthermore, only one study adds random process noise in addition to measurement noise. Therefore, it is difficult to comment on the ability of the approaches to perform quick detection and diagnosis whilst being robust to noise and modelling uncertainties.

The probabilistic nature of DBN-based PD approaches does, however, provide estimation of classification error in fault detection, i.e. by taking into account different types of error by explicitly modelling uncertainty. This allows for tuning of the approach, which facilitates better decision-making under uncertainty. Therefore, it is possible that a DBN-based PD approach may be robust to noise and modelling uncertainties despite this not being made apparent in the studies discussed in this chapter.

If this is true and faults may be reliably detected, it would be valuable if the approach provided explanation facility for fault propagation from the root cause to its observed symptoms. Since DBN models explicitly model causal relationships, they should be able to provide explanation facility via some visualisation of the DBN CPDs similar to that in Yu and Rashid (2013).[2]

Another useful feature of the model-based nature of the approaches is that they allow for the identification of multiple simultaneous faults. This is a powerful capability which can be used to quickly diagnose multiple faults instead of detecting and diagnosing each fault individually over a longer period of time.

In conclusion, DBN-based PD approaches show promise for use as fault diagnosis tools since they possess most of the desired characteristics in Section 2.2. Furthermore, these approaches may also be combined with data-driven techniques for fast and reliable fault detection as presented in Yu and Rashid (2013). However, since the set of possible root causes is limited to observed PVs in the approach by Yu and Rashid (2013), we rather choose to extend the approach in Lerner (2002) in this thesis. This affords us the ability to define our set of possible root causes from the ground up and design the diagnosis approach around them while maintaining a simpler approach (i.e. single stage) than those in Roychoudhury *et al.* (2006, 2008, 2009, 2010).

[2]Note we do not investigate explanation facility in this thesis. However future research may investigate this based on the viability of the PD approach.

Chapter 5

Research Methodology

It is difficult to design and test fault diagnosis strategies using real process data. In order to collect even the simplest form of real process data (i.e. containing no faults), one would first have to solve the fault diagnosis problem. Simulations of processes provide an alternative source of data to design and test fault diagnosis strategies and are used in most literature as discussed in Chapter 4. Although such simulations may not necessarily depict real processes to high degrees of accuracy, these allow their users to manipulate many more (simulated) PVs and quickly generate large quantities of useful data. In theory, a well-designed process simulation with a reasonably accurate model should be able to produce fault diagnosis strategies that are scalable to the real process.

Given the degrees of freedom in simulation, one must not lose sight of the real process and simulate unrealistic process conditions. This chapter discusses the methods used in this thesis to develop process simulations for benchmarking the application of fault diagnosis approaches. In particular, this research considers two case studies, both of which have different configurations of increasing complexity. The first case study is used for evaluation of the proposed DBN-based PD approach of Chapter 3 and comparison thereof with the MSPM approach of Chapter 2 (Section 2.1). The second case study is used to analyse scalability of the PD approach. Penultimately, both fault diagnosis approaches are briefly summarised in this chapter, with additional attention given to challenges in implementation of the PD approach. The chapter is concluded with a discussion on testing methodology for fault diagnosis performance evaluation and comparison.

5.1 Data generation

Ordinary differential equation solvers enable simulation of the time evolution of mathematical process models. Simulink™ in MATLAB® offers a number of ordinary differential equation solvers and a user-friendly environment to build process simulations. Its graphical interface allows a user to create various perturbations to

a process model, such as disturbances and faults, and observe the response of that process model as time-series data.

Perturbing a mathematical process model with a single fault is useful for testing isolability of a fault diagnosis algorithm, but does not provide information about the robustness of that algorithm. This section is concerned with building realistic process simulations which feature sensor noise, process noise (i.e. disturbance perturbations), and regulatory control systems.

5.1.1 Sensors

In practice, sensors provide an interface for observing the PV values of particular process units and streams. Sensor noise is typically high-frequency noise caused by the fast sampling rates and the propagation of errors in measurement principles. This combination causes sensor readings to be noisy, but generally centred around the actual PV value.

In simulations, this is mimicked by considering, conceptually, a mathematical process model as the physical process whose actual PV values may only be observed using simulated sensors. These sensors are assumed to have a fixed sampling rate and add white noise to their actual PV values as shown in Figure 5.1. Although the variance of the white noise (i.e. sensor variance) depends on the type of sensor being modelled (since some are more precise than others), sensor variance is typically modelled as being a fraction of a PV's nominal value for each sensor.

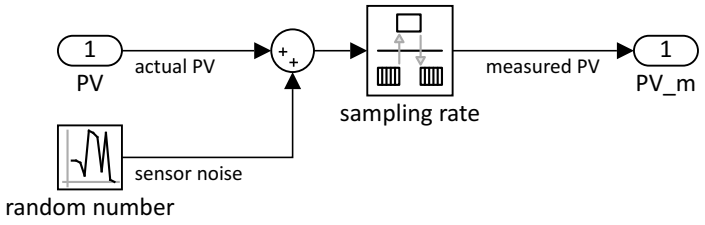


Figure 5.1: Sensor implementation in Simulink™ .

5.1.2 Disturbances

Disturbances are uncontrollable process inputs, such as varying feed conditions (e.g. in flow, temperature, pressure, composition, particle size distribution), or variation in ambient weather conditions (e.g. rain affecting open top process units such as flotation cells). In practice, disturbances create a "process" noise that is lower in frequency than the sensor noise but that directly influence nominal operation of the physical process; hence, regulatory control systems are put in place to mitigate the effect of disturbances on product quantity and quality.

In simulations, disturbances should be treated as part of the actual process or at least as uncontrollable inputs into the process. Varying feed conditions are possibly the most common type of disturbance to a process; typically this type of disturbance varies slowly between known upper and lower bounds. Miskin (2016) models this variation as a one-dimensional random walk (Lawler and Limic, 2010), a special kind of autoregressive function which preserves the inertia of variations in a disturbance PV. The approach of Miskin (2016) is further discussed here and used to model disturbances in this thesis.

Consider a disturbance PV v which varies in time according to a variable l. l and $-l$ describe the gradients by which v increases or decreases between two discrete time steps (i.e. the sampling rate of the random walk). A uniformly distributed random number between 0 and 1, x_i, is generated for every time step i which produces a combination of gradients, s_i, according to Equation 5.1.

$$s_i = \begin{cases} +l & x_i < 0.5 \\ -l & x_i \geq 0.5 \end{cases} \tag{5.1}$$

Each s_i indicates whether v increases or decreases between two time steps. The time-dependent value of v is therefore a numerical integration of all s_i up to time t according to Equation 5.2.

$$v(t) = \sum_{i=1}^{t} s_i \Delta t + v(0) \tag{5.2}$$

where $v(0)$ is the initial value of v.

It remains to bound the random walk of v with an upper and lower limit such that $v(t)$ never increases above the upper limit of v and never decreases below the lower limit of v. This may be done by checking that $v(t)$ is indeed between the bounds and if not, recomputing $v(t)$ with an opposite gradient for the latest s_i.

Figure 5.2 shows the implementation of a random walk in Simulink™ .

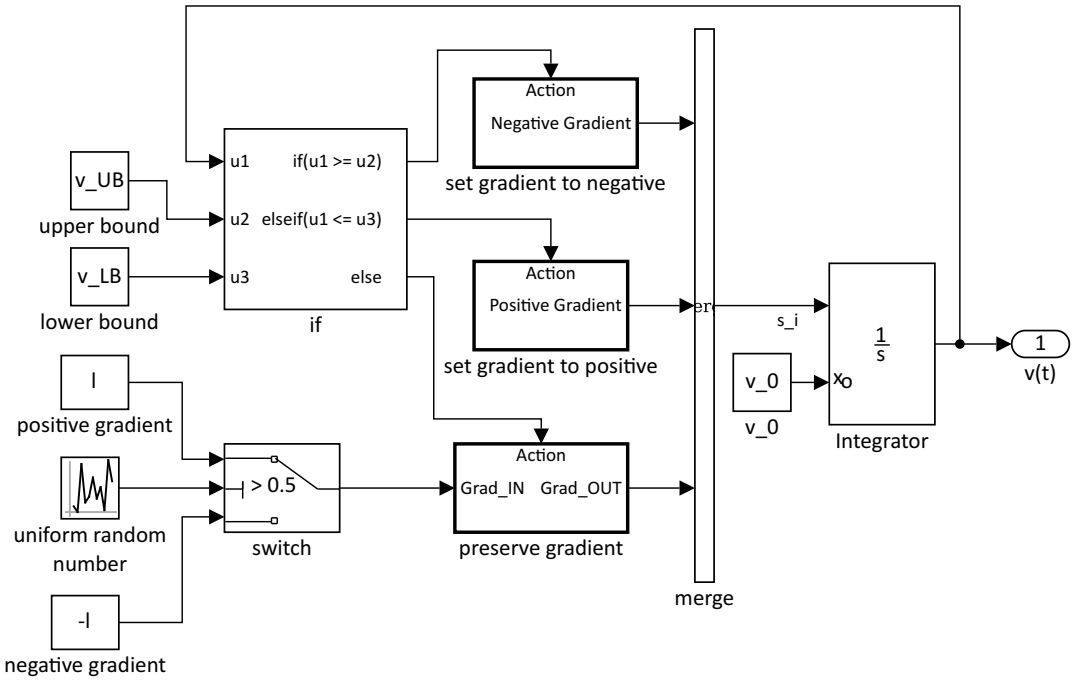


Figure 5.2: One-dimensional random walk implementation in Simulink™ .

5.1.3 Regulatory control

Regulatory control systems mitigate the effects of disturbances on processes and also allow set point tracking. The simplest form of such systems uses feedback control which takes action based on the (observed) difference between current process conditions and the desired (set point) operating conditions.

In practice, processes typically use multiple controllers from the family of Proportional-Integral-Derivative (PID) controllers for regulatory control (Marlin, 2000). Equation 5.3 shows the Proportional-Integral (PI) version of such a controller, which *manipulates* an input PV, MV, in order to *control* an output PV, CV, such that the difference between $CV(t)$ and the *set point* value of CV, SP, is minimal.

$$\begin{aligned} E(t) &\equiv SP(t) - CV(t) \\ MV(t) &= K_c \left(E(t) + \frac{1}{T_I} \int_0^t E(t')\,\mathrm{d}t' \right) + MV(0) \end{aligned} \tag{5.3}$$

In these equations the controller gain K_c and integral time constant T_I tune the response of the PI controller, and $MV(0)$ is the initial value of MV. Typically, a robust controller is expected to maintain a small (if not zero) error, $E(t)$, without large variation in $MV(t)$.

Controllers are treated as separate from the physical process. Figure 5.3 shows

the implementation of a PI controller in Simulink™ . Note that $MV(t)$ is bounded by upper and lower saturation limits in the simulation. This ensures that the controller calculated values for $MV(t)$ do not exceed the physical limitations of that MV. For example, this bounds position of control valves between fully open and fully closed.[1]

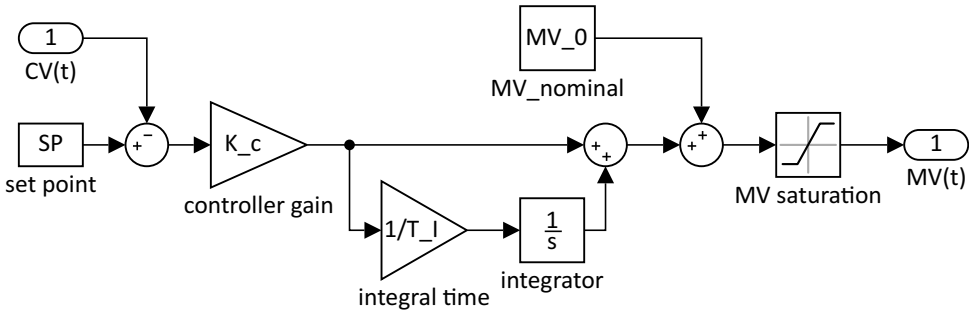


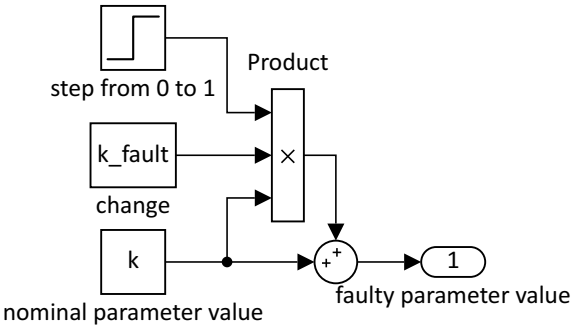
Figure 5.3: PI controller implementation in Simulink™ .

5.1.4 Faults

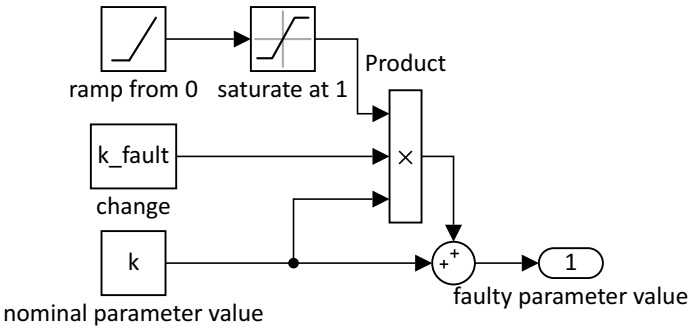
The class of a fault dictates how it is implemented in a process simulation. Recall from Section 2.2.2.2 that there are generally four fault classes: (1) gross parameter changes in a model, (2) structural changes, (3) malfunctioning sensors, and (4) malfunctioning actuators. The first two fault classes manifest within the physical process (i.e. modifying the mathematical process model), whilst the third and fourth fault classes manifest within the sensor models and controllers respectively.

Consider a pipe blockage in the two-tank system shown in Figure 5.6 in Section 5.2.1 (a class 1 fault). This fault may be simulated by decreasing one of k_1, k_2, or k_{12}, causing reduced flow through the associated pipe. The manner in which the fault manifests (i.e. abrupt or incipient) is defined by how that change occurs as shown in Figure 5.4: as a step for the abrupt case (Figure 5.4a) or as a ramp for the incipient case (Figure 5.4b).

[1] Note that saturation of a manipulated PV does not necessarily imply abnormal process behaviour.



(a) Abrupt case



(b) Incipient case

Figure 5.4: Parameter change fault implementation in Simulink™ .

Now consider a class 3 fault: a sensor bias fault which manifests abruptly inside a sensor model as shown in Figure 5.5. Notice that, in contrast to Figure 5.1, the actual value of the PV being measured has a constant bias fraction added to it before white noise and sensor sampling rate are taken into account.

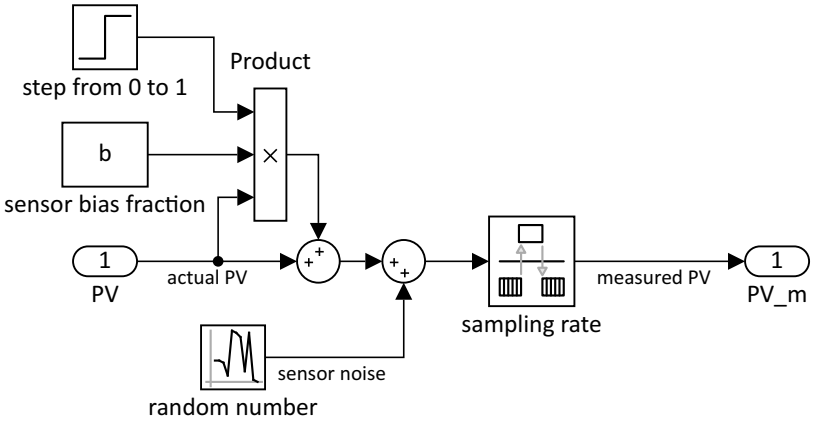


Figure 5.5: Sensor fault implementation in Simulink™ .

The implementation of faults in the manner shown in Figures 5.4 and 5.5 enable these faults to manifest at user-specified times, allowing for time-series data generation containing both NOC and fault conditions (after manifestation of a single fault – multiple simultaneous fault manifestations are not considered in this thesis).[2]

5.2 Case studies

This section discusses the case studies considered in this thesis: two and five tanks in series. Both case studies were inspired by Lerner *et al.* (2000), but our versions include additional aspects, namely regulatory control and recycle structures.

5.2.1 Two tanks in series

Description Two tanks are connected by a pipe between their bottoms as shown in Figure 5.6. Both tanks are open to the atmosphere and drain out their bottoms naturally through a flow constriction. The first tank receives a feed flow of water.

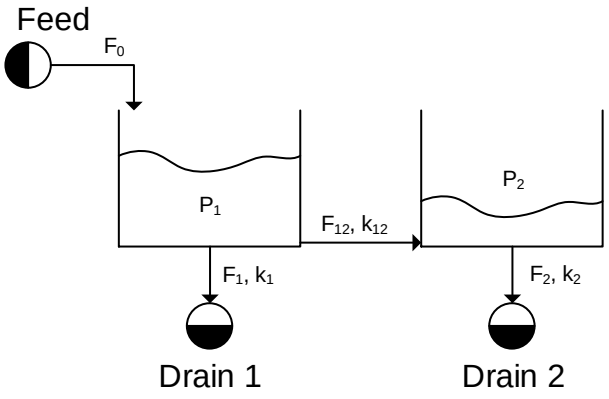


Figure 5.6: Two tanks in series.

Process model Appendix B.1 shows how the two-tank system is modelled, obtaining the Governing Equations 5.4 and the Constitutive Equations 5.5.

$$\begin{aligned} \frac{\mathrm{d}P_1}{\mathrm{d}t} &= \frac{\rho g}{A_1}\left(F_0 - F_1 - F_{12}\right) \\ \frac{\mathrm{d}P_2}{\mathrm{d}t} &= \frac{\rho g}{A_2}\left(F_{12} - F_2\right) \end{aligned} \tag{5.4}$$

[2]Note that our simulations support multiple simultaneous fault manifestations, however we leave this topic for future work.

$$F_1 = k_1\sqrt{P_1} \quad \text{and} \quad F_2 = k_2\sqrt{P_2} \quad \text{and} \quad F_{12} = k_{12}\sqrt{|P_1 - P_2|} \times \text{sign}(P_1 - P_2) \tag{5.5}$$

Table 5.1 shows the initial and nominal model parameters which were used to simulate this system. These values were chosen such that the system is initialised with its steady-state values.

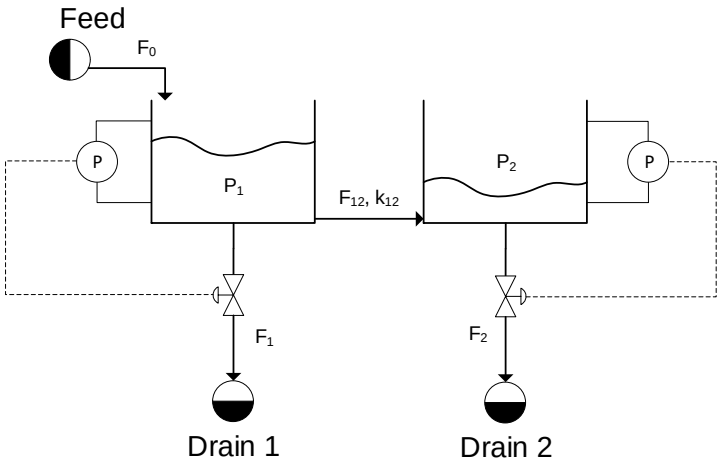
Table 5.1: Model parameters and nominal PV values for two-tank system.

Symbol	Meaning	Unit	Nominal value
A	Cross-sectional area	m^2	$A_1 = A_2 = 1$
k	Constriction resistance	$\text{m}^{3.5} \cdot \text{kg}^{-0.5}$	$k_1 = k_2 = 1\text{e-}5$
			$k_{12} = 2\text{e-}5$
F	Flow	$\text{m}^3 \cdot \text{s}^{-1}$	$F_0 = 1.70\text{e-}3$
			$F_1 = 0.90\text{e-}3$
			$F_{12} = 0.80\text{e-}3$
			$F_2 = 0.80\text{e-}3$
P	Pressure	Pa	$P_1 = 8053$
			$P_2 = 6442$
ρ	Water density	$\text{kg} \cdot \text{m}^{-3}$	$\rho = 1000$
g	Gravitational acceleration	$\text{m} \cdot \text{s}^{-2}$	$g = 9.81$

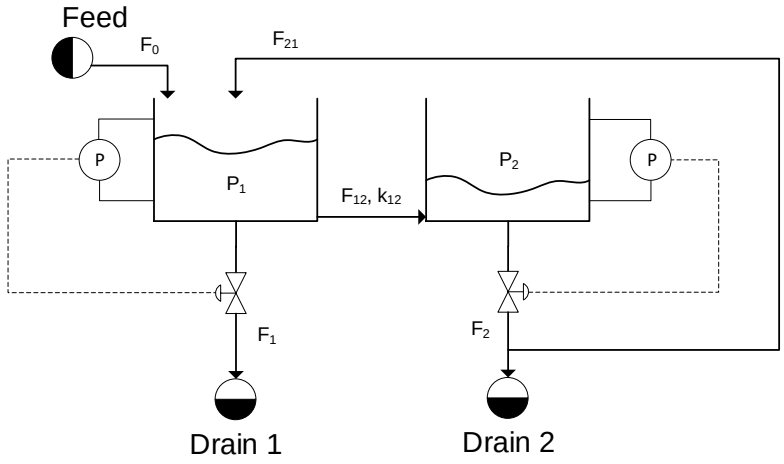
Case study motivation Most industrial processes in chemical engineering and mineral processing involve hydraulic processes. The two-tank system is a simplistic representation of such processes and is a good case study to show proof of concept and trouble shoot initial implementation of the PD approach.

Configurations Complexity of the case study was varied by considering four configurations: (sN) sensor noise only, no regulatory control, no recycle (see Figure 5.6);[3] (N) process and sensor noise, no regulatory control, no recycle (see Figure 5.6); (NC) process and sensor noise, tank pressure control, no recycle (see Figure 5.7a); and (NCR) process and sensor noise, tank pressure control, 80% of F_2 recycled back to the first tank (see Figure 5.7b). The configurations were chosen such that the robustness of the fault diagnosis approaches to noise and increasing process complexity could be tested.

[3] Note that the sN configuration is purposely unrealistic and was tested for the intention of evaluating isolability in fault diagnosis performance.



(a) [NC] System with regulatory control.



(b) [NCR] System with recycle and regulatory control.

Figure 5.7: Two-tank system in configurations NC and NCR.

Sensors were sampled at a constant rate of 1 Hz and sensor noise was implemented by adding Gaussian noise with zero mean and a standard deviation equal to 1.5% of the nominal measured value, to each measured PV. Note that it is assumed that sensor bias faults do not affect the noise distributions of the sensors.

Two simple pressure-flow feedback control loops were introduced under the NC configuration, as shown in Figure 5.7a, to regulate the pressure within each tank. The regulatory control system was tuned to have a conservative response for re-use under the NCR configuration. The tuning parameters, i.e. K_c and T_I for Equation 5.6, for

each of the two control loops are given in Table 5.2.[4]

$$\begin{aligned} E(t) &= P_{\text{nominal}} - P(t) \\ F(t) &= K_c \left(E(t) + \frac{1}{T_I} \int_0^t E(t')\,\mathrm{d}t' \right) + F_{\text{nominal}} \end{aligned} \tag{5.6}$$

In these equations P_{nominal} and $P(t)$ are the set point and measured value of one tank's actual pressure and F_{nominal} and $F(t)$ are the nominal and manipulated value of that tank's flow out of its drain, respectively.

Table 5.2: Controller parameters for the two-tank system.

Symbol	Meaning	Unit	Value
K_c	Controller gain	$\mathrm{m}^3 \cdot \mathrm{s}^{-1} \cdot \mathrm{Pa}^{-1}$	$K_{c1} = -2.53\text{e-}7$
			$K_{c2} = -2.32\text{e-}7$
T_I	Integral time constant	s	$T_{I1} = 128.75$
			$T_{I2} = 126.25$

Process noise was implemented as a random walk function for F_0 with bounds 10% above and below its nominal value and a gradient of 0.0005 $\mathrm{m}^3 \cdot \mathrm{s}^{-2}$ sampled every 2 minutes.

The NCR configuration had the same implementation as the NC configuration for process and sensor noise, and regulatory control. Recycling 80% of F_2 back to the first tank caused the steady-state PV values to change. All pressures and flows maintained the same nominal values as in the other configurations except for F_1 which was changed to 1.54e–3 $\mathrm{m}^3 \cdot \mathrm{s}^{-1}$.

Fault selections Four faults were tested for each configuration: (aConn) abrupt partial blockage in the connecting pipe; (iConn) incipient partial blockage in the connecting pipe; (P1s) tank 1 pressure sensor bias; and (P2s) tank 2 pressure sensor bias.

aConn and iConn were implemented by changing the constriction resistance of the connecting pipe (see Figure 5.4). At its maximum, the partial blockage fault caused a 80% decrease in k_{12}. This was implemented as a step input for the abrupt case and ramped linearly with gradient 0.17 $\% \cdot \mathrm{s}^{-1}$ for the incipient case.

[4] Note that the values shown for K_c are small because the flow rates (i.e. manipulated PVs) are small while the pressures (i.e. controlled PVs) are large, thus a small change in a controlled PV should produce a much smaller change in the appropriate manipulated PV.

P1s and P2s were implemented as abrupt sensor bias faults (see Figure 5.5). Each fault caused a -20% bias (prior to noise) in the readings of the pressure sensor of the relevant tank.

5.2.2 Five tanks in series

Description Similar to the two-tank system of Section 5.2.1, five tanks are connected by pipes between their bottoms as shown in Figure 5.8. All tanks are open to the atmosphere and drain out their bottoms naturally through flow constrictions. The first tank receives a feed flow of water.

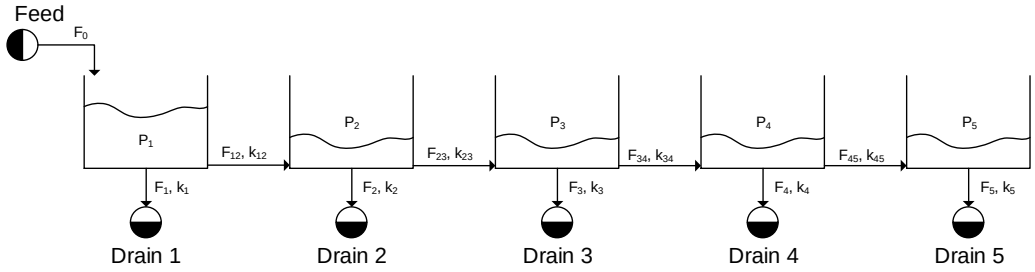


Figure 5.8: Five tanks in series.

Process model The five-tank system has the Governing Equations 5.7 and the Constitutive Equations 5.8 and nominal model parameters listed in Table 5.3.

$$\begin{aligned}
\frac{\mathrm{d}P_1}{\mathrm{d}t} &= \frac{\rho g}{A_1}\left(F_0 - F_1 - F_{12}\right) \\
\frac{\mathrm{d}P_i}{\mathrm{d}t} &= \frac{\rho g}{A_i}\left(F_{i-1,i} - F_i - F_{i,i+1}\right) \quad i = 2, \ldots, 4 \\
\frac{\mathrm{d}P_5}{\mathrm{d}t} &= \frac{\rho g}{A_5}\left(F_{45} - F_5\right)
\end{aligned} \tag{5.7}$$

$$\begin{aligned}
F_j &= k_j\sqrt{P_j}, \quad j = 1, \ldots, 5 \\
F_{j,j+1} &= k_{j,j+1}\sqrt{|P_j - P_{j+1}|} \times \operatorname{sign}(P_j - P_{j+1}), \quad j = 1, \ldots, 4
\end{aligned} \tag{5.8}$$

Table 5.3: Nominal values of model parameters for the five-tank system.

Symbol	Meaning	Unit	Nominal value
A	Cross-sectional area	m^2	$A_i = 1, \quad i = 1, \ldots, 5$
k	Constriction resistance	$\mathrm{m}^{3.5} \cdot \mathrm{kg}^{-0.5}$	$k_i = 1\text{e-}5, \quad i = 1, \ldots, 5$
			$k_{j,j+1} = 2\text{e-}5, \quad j = 1, \ldots, 4$
F	Flow	$\mathrm{m}^3 \cdot \mathrm{s}^{-1}$	$F_0 = 1.70\text{e-}3$
			$F_1 = 0.66\text{e-}3$
			$F_2 = 0.41\text{e-}3$
			$F_3 = 0.26\text{e-}3$
			$F_4 = 0.19\text{e-}3$
			$F_5 = 0.17\text{e-}3$
			$F_{12} = 1.04\text{e-}3$
			$F_{23} = 0.63\text{e-}3$
			$F_{34} = 0.36\text{e-}3$
			$F_{45} = 0.17\text{e-}3$
P	Pressure	Pa	$P_1 = 4379$
			$P_2 = 1684$
			$P_3 = 699$
			$P_4 = 368$
			$P_5 = 295$
ρ	Water density	$\mathrm{kg} \cdot \mathrm{m}^{-3}$	$\rho = 1000$
g	Gravitational acceleration	$\mathrm{m} \cdot \mathrm{s}^{-2}$	$g = 9.81$

Motivation The five-tank system is a larger-scale version of the two-tank system, consisting of more PVs and subsystems. It is an ideal system to test the ability of the PD approach to scale to larger processes while still being able to relate results to the smaller two-tank system.

Configurations Three configurations for the five-tank system were considered, namely: (N) process and sensor noise, no regulatory control, no recycle; (NC) process and sensor noise, tank pressure control, no recycle; and (NR) process and sensor noise, no regulatory control, 80% of the drainage from the fifth tank recycled back to the first tank. Regulatory control for the NC configuration was implemented in the same manner as in the two-tank system, i.e. where each control loop controls a tank's pressure by manipulating drain outflow from that tank.

The tuning parameters for the five pressure-flow PI control loops are given in Table 5.4.

Table 5.4: Controller parameters for the five-tank system.

Symbol	Meaning	Unit	Value
K_c	Controller gain	$\mathrm{m}^3 \cdot \mathrm{s}^{-1} \cdot \mathrm{Pa}^{-1}$	$K_{c1} = -2.53\text{e-}7$ $K_{cn} = -2.32\text{e-}7, n = 2, \ldots, 5$
T_I	Integral time constant	s	$T_{I1} = 128.75$ $T_{In} = 126.25, n = 2, \ldots, 5$

Fault selection Only one fault was considered, namely an abrupt partial blockage in the connecting pipe between tanks 2 and 3. The implementation of the fault was the similar to that in the two-tank system, but instead caused a 45% decrease in k_{23}.

5.3 Tested fault diagnosis approaches

This section provides a visual overview of both the MSPM approach and the PD approach tested in this thesis. Specific settings and implementation methods are highlighted as well.

5.3.1 Overview of MSPM approach

The MSPM approach (Figure 5.9) is a relatively simple data-driven approach. As detailed in Section 2.1, it captures the structure of variation in the data under NOC and then compares new samples to this structure, detecting fault conditions when the deviation of a sample is unusual for those under NOC .

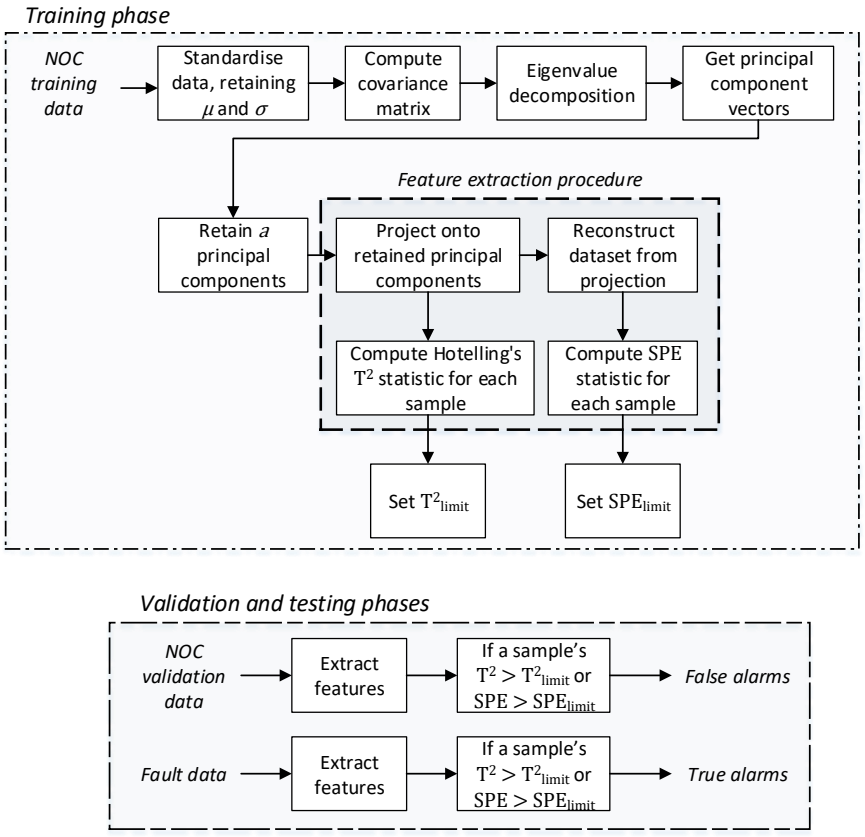


Figure 5.9: An overview of the MSPM approach to fault diagnosis.

Since MSPM is a data-driven approach, its modelling requirements are not as high as those of the PD approach. Furthermore, it only requires the specification of one hyper-parameter (not including fault detection limits) for training, i.e. the minimum percentage of variance explained by the retained PC – this was set to 70% in this thesis.

The parameter is only required for the training phase of the approach, which determines the number of retained PC axes. Once the training phase is complete, the feature extraction procedure is used to extract features from new samples and the values of their T^2 and SPE statistics may be compared to the respective limits obtained from the training data. Note detection limits for these statistics are discussed later in Section 5.4.

5.3.2 Overview of PD approach

In contrast to the MSPM approach, the PD approach, summarised in Figure 5.10, is a model-based fault diagnosis approach.

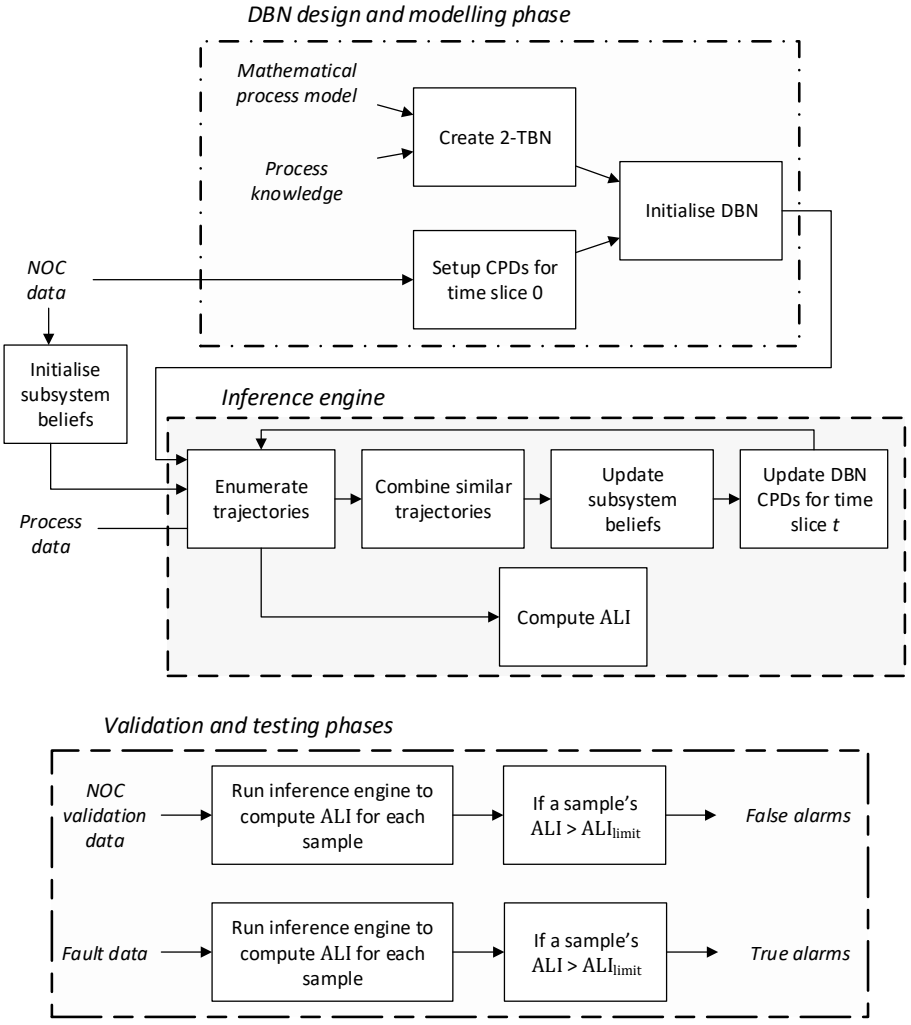


Figure 5.10: An overview the of PD approach to fault diagnosis.

Note that there is no explicit training phase for this approach, instead a DBN design and modelling phase replaces it.[5] This phase concerns the creation of the internal model of the approach and is required for the inference engine. The reader is referred back to Section 3.4 for particular DBN design and modelling choices made in this work.

Furthermore, as part of the design of the DBN, one must also specify how to distribute the process state RVs among the different subsystems. As discussed in Section 3.3.2, each subsystem must contain a unique subset of these RVs (i.e. not overlapping with other subsystems) and union of the RVs in all the subsystems should be the same as all the process state RVs. In this thesis and since our case studies only involve a number of tanks connected in series, each subsystem corresponds to a tank in each of our case studies. The PVs associated with a tank are the inflow to that tank, the pressure within that tank, and the outflow from that tank excluding outflow through a connecting pipe or recycle stream. In addition, the sensor model RVs, as well as any RVs pertaining to fault models, associated with a PV are also included in the same subsystem. Lastly, the RVs pertaining to a control loop are included in the subsystem containing the controlled PV for that loop. Table 5.5 illustrates this approach to defining subsystems for the two-tank system.

Table 5.5: PD approach: Illustrated subsystem decomposition for the two-tank system. The subscripts $_m$ and $_b$ indicate measured (observed) RVs and bias terms, respectively. Note that some RVs subsume model parameters.

Tank 1	Tank 2
F_0	F_{12}
Status $F_{0,m}$	Status $F_{12,m}$
$F_{0,m}$	$F_{12,m}$
$F_{0,b}$	$F_{12,b}$
P_1	P_2
Status $P_{1,m}$	Status $P_{2,m}$
$P_{1,m}$	$P_{2,m}$
$P_{1,b}$	$P_{2,b}$
$S_{\text{acc},1}$	$S_{\text{acc},2}$
F_1	F_2
Status $F_{1,m}$	Status $F_{2,m}$
$F_{1,m}$	$F_{2,m}$
$F_{1,b}$	$F_{2,b}$

The inference engine uses the DBN and initial beliefs to predict process behaviour and update its beliefs (decomposed into subsystems) with respect to the discrepancies between its predictions and observed sensor readings. It also computes an abnormality

[5] Note that robust estimation of CPDs from data as well as DBN structure learning are topics for future research and are beyond the scope of this thesis.

likelihood index (ALI) which is the probability of the process being in an abnormal operating state. This procedure is repeated for each sample in the data set, but instead using the updated beliefs to predict process behaviour.

Unlike the feature extraction procedure of the MSPM approach, the inference engine updates its beliefs after each new sample. Therefore, new samples are not considered in isolation but instead each sample adds more information to the beliefs, further fine-tuning the CPDs in the DBN to ensure reliable model predictions over time. For this reason, the validation and testing data are passed through the inference engine as a single data set instead of as separate, time-independent samples as with the MSPM approach.

Hyper-parameters for the PD approach include: c and K as defined in Section 3.2.2 on page 59, and $T_{\max}$ and $\Pr_T$ in Section 3.3.1 on page 63. To recap, these parameters determine: the threshold for the similarity metric when comparing transitions (c), the maximum number of hypotheses permitted per subsystem (K), the maximum number of transitions that may be enumerated from an approximated process state ($T_{\max}$), and the cumulative *a priori* probability threshold sufficient for enumeration ($\Pr_T$).

$T_{\max}$ and $\Pr_T$ were chosen to be fixed in this work and was not further investigated for optimality. Their values were set to $T_{\max} = 10000$ and $\Pr_T = 1 - 1e\text{–}12$. These choices ensured sufficient transitions would be enumerated from each time slice for hypothesis generation, without enumerating all possible transitions.

The values of c and K, on the other hand, were chosen based on findings presented in Lerner (2002). Lerner (2002) observed more stable process state tracking performance for a value of $c = 1$ versus $c = 50$ when varying the value of K between 1 and 12 for a small system with no subsystem decomposition. Therefore, the value of c was chosen to be 1. The value of K was chosen to be 15 (per subsystem) to ensure sufficient consideration of different and dissimilar hypotheses during belief state approximation.

These settings were used for the evaluation of the fault diagnosis performance of the PD approach and comparison thereof with the MSPM approach using the two-tank system. It was also investigated how different choices of values for c and K, namely $c = \{0, 0.0001, 1, 10000, \infty\}$ and $K = \{1, 5, 100\}$, affect fault diagnosis performance using the two-tank system. Note that later tests for scaling the PD approach up to the five-tank system used $c = 1$ and $K = 5$ due to higher memory usage associated with higher values of K especially for larger processes, i.e. with more components which are modelled in the DBN.

5.3.3 Implementation challenges and solutions for PD approach

Note that a number of challenges were faced during our implementation of the PD approach in this work. Some noteworthy challenges and solutions which may affect the description of the approach in Chapter 3 are discussed here. Additionally, some testing was performed on a simplified version (with respect to the DBN model) of the two-tank system in Section 6.1 to verify our implementation of the PD approach.

Non-linear CPDs:
A major issue with respect to implementation of the PD approach was the lack of support for non-linear relationships between continuous RV nodes in the BNT (Murphy, 2002). This issue was resolved using a particular linearisation method, based on the unscented transform, described in Section 3.4.2.

Numerical issues:
The BNT performs inference calculations using canonical forms for probability distributions. These forms specify all probability distributions in the form $\exp\left(Q(\mathbf{x})\right)$ where $Q(\mathbf{x})$ is some function unique to the particular type of distribution being represented, namely $Q(\mathbf{x})$, for Gaussians, is a quadratic function. The use of these canonical forms in the BNT meant that probability distributions often had to be converted to and fro between the moment form (as in Section 2.3.2) and canonical form. In particular, this causes issues for Gaussian probability distributions due to limited precision in floating-point values and relatively ill-conditioned covariance matrices.

It was eventually found that most of these issues were caused by invalid covariance matrices resulting from calculations performed by the BNT. In particular, these matrices would be non-symmetric and not positive semi-definite. In these cases where it was necessary to use the moment form (such as when combining transitions), a function by John D'Errico from the MathWorks Exchange, called `NearestSPD`[6], was used to find the symmetric positive semi-definite matrix nearest in Frobenius norm to the invalid covariance matrix (Higham, 1988). This was also supplemented by another function which would perturb the diagonal of the covariance matrix slightly in order to improve its condition number when `NearestSPD` would fail.

DBN design challenges:
As discussed in Lerner (2002) the most challenging part of the PD approach is DBN modelling. Anomalous results may be frequent due to bad or incorrect DBN design and specification because the PD approach relies heavily on model predictions. The solution to this problem is to use simple DBN designs with few tunable parameters

[6] `NearestSPD` is available at www.mathworks.com/matlabcentral/fileexchange/42885-nearestspd.

such as presented in Section 3.4. Although it is possible, in our case, to use one DBN design for multiple different configurations (i.e. by over designing) of the same system (such as for the two-tank system), doing so results in unnecessary complication of the DBN design. This not only makes it more difficult to identify modelling issues, but also causes simpler configurations to require more computational resources than necessary. Therefore, this thesis uses different DBN designs for different configurations of the same system.

State tracking challenges:
A solution that arose during debugging of modelling issues was to separate the transition from normal operating mode to normal operating mode from all other transitions. This meant that this normal operating mode to normal operating mode transition would always be enumerated and never combined with any other transitions, ensuring that the normal operating mode hypothesis would be present in each time slice even its likelihood was extremely low. Doing this avoids scenarios where the normal operating mode to normal operating mode transition is not enumerated for every time slice due the process state being tracked incorrectly. This subsequently reduces the likelihood of creating hypotheses which are not supported by, or have little support from, the observations of process behaviour. Note that, in this thesis, this normal operating mode hypothesis does not contribute to the hyper-parameter K – thus, including this hypothesis, $K+1$ hypotheses are permitted for each subsystem.

Furthermore, note that in the event of incorrect process state tracking leading to no support from observations for any hypothesis, our implementation invokes a DBN reset. This reset reinitialises the DBN to the initial state at time slice 0 and then attempts to resume process state tracking. If the process state still cannot be tracked well enough after the reset or if the DBN needs to be reset repeatedly, one may conclude that there is likely an issue with the DBN model which needs to be further investigated and fixed.[7]

Omission of smoothing:
Smoothing considers multiple samples in a time frame of a particular size (i.e. a window) and processes all of them before producing a result. Typically this result is a decision for a time slice a few steps behind the current observation, enabling the fault diagnosis approach to "look ahead" before producing diagnosis results for a particular time slice. Although Lerner (2002) incorporates this technique in their approach, we do not do this because the computational time required to process a sample was found to be relatively high, i.e. on the order of seconds, without smoothing and this would increase if smoothing were implemented.

Testing of concurrent faults:
The manifestation of multiple faults in a sequence of fault data describes a case study

[7] Note this was not an issue in our experiments.

which contains concurrent faults. Although the PD approach has the ability to detect and diagnose concurrent faults, we do not explore this in this thesis. We explicitly omit this testing because there remains a number of issues with single fault case studies which need to be resolved before moving on to diagnosis of concurrent faults.

5.4 Testing methodology

This section presents additional details about the methods used for testing the fault diagnosis performance of the MSPM and PD approach.

Computation details:
All testing was performed using an Intel Xeon X5680 3.33 GHz processor with 16 GB RAM running 64 bit Windows Server 2008 R2 Enterprise (SP1). Code was implemented in MATLAB® R2017a and will be made available at a later stage under an open source license. Included in the open source code will be the Simulink™ files for the case studies, the MSPM approach, the PD approach, tools for computing fault diagnosis performance, and reproducibility scripts to reproduce key results presented in Chapter 6.

Data characteristics:
The simulation data for each fault under each configuration was generated as a continuous segment representing 30 minutes in real-time. In each data set a fault would manifest after 15 minutes, producing even amounts of NOC and fault data. Since the sensor sampling rates were set to 1Hz, each data set contained 1800 samples with a fault manifesting at the 900th sample.

The NOC data was identical for all faults under each configuration and was split in a 60:40 ratio for the training and validation phases of the MSPM approach. The training portion of the split was used to determine the initial BN CPDs for the PD approach.

Full observability was assumed, therefore all measured PVs were observed for both the MSPM and PD approaches. In this work, these were all pressures and flows for the two case studies in Section 5.2.

Fault detection limit selection:
Fault detection limits for both the MSPM approach and PD approach were based on 99th percentiles. This was done to emulate realistic detection limit selection based on availability of *a priori* knowledge.

In the case of the MSPM approach, the user typically only has available the scores for the monitoring statistics obtained from training data. Therefore, for this approach, fault detection limits were set to the 99th percentile value of T^2_{NOC} for T^2_{limit} and the

99^{th} percentile value of SPE_{NOC} for SPE_{limit}.

In the case of the PD approach, although a similar approach to choosing a fault detection limit may be used, doing so ignores important characteristics for the nature of the abnormality likelihood index. Recall that ALI is a likelihood for abnormality and is known to range from 0 to 1, such that 1 indicates a reasonable certainty for abnormal behaviour. Furthermore, realistic process data typically does contain minor deviations from NOC (i.e. impulses) which could be considered abnormal if ALI_{limit} chosen to be a low value closer to 0. Therefore, it is better to choose ALI_{limit} such that it considers allowable deviation from NOC and does not spuriously report minor deviations from NOC as the result of a fault. In this thesis, ALI_{limit} was therefore chosen to be 0.99.

Fault detection performance metrics:

In Section 2.2.3, AUC was presented as a good performance metric for evaluating fault diagnosis performance regardless of detection limit selection. This metric is useful for evaluating the overall ability of a fault diagnosis approach to discriminate between normal and abnormal process behaviour. However, more detailed fault detection performance metrics can be evaluated once detection limits are chosen.

Among these metrics are FAR, TAR, and F1 score also discussed in Section 2.2.3. Since F1 score essentially combines the information contained in FAR and TAR into a single metric it is very useful for summarising this detail of fault detection performance. Therefore, F1 score is used in this thesis to evaluate the ability of a fault diagnosis approach to discriminate between normal and abnormal process behaviour given a particular set of detection limits.

It is important to note that each alarm reported by a fault diagnosis approach should not immediately be treated as detection of fault conditions. Especially in the case of the MSPM approach, because of the way in which detection limits were selected, it is possible that a sample and several subsequent samples obtained during normal process behaviour may trigger the approach to report an alarm for the one sample and not for the others. In these cases it would be ineffective to attempt to diagnose fault conditions simply because there would be no fault to diagnose. This can be avoided by instead treating consecutive alarm reports as indicative of fault conditions and only recognising the detection of fault conditions after a sufficient number of consecutive alarms had been reported.

In this thesis that number was chosen to be 10. This meant that fault conditions would only be detected when alarms were reported for at least 10 consecutive samples. Therefore, the minimum delay in detection of fault conditions in our tests was 10 samples. However, note that these 10 samples are excluded from values when discussing detection delays in Chapter 6 for ease of use. Therefore, a detection delay of 0 samples is indicative of immediate fault detection since 10 consecutive samples would in fact report alarms after manifestation of the fault.

Root cause analysis performance metrics:

We are unaware of any qualitative root cause analysis performance metrics, but make an attempt to quantify the success of root cause analysis in terms of some observations. In particular, if the root cause of a fault was identified among possible fault symptoms on a contribution plot (see Section 2.1.4) or component ALI plot (see Section 3.2.3), then one can make certain quantitative observations about the nature of the root cause analysis result. This includes the number of fault symptoms identified alongside the root cause as well as the ranking of the root cause in terms of its relative contribution or component ALI compared to the values for the identified fault symptoms.

Note creation of contribution plots and component ALI plots considers the average relative contributions and average component ALIs for over a particular number of samples. These samples are the first 100 samples after the detection of fault conditions; this includes the 10 consecutive samples which triggered detection of fault conditions.

Conclusion:

This chapter sets the stage for evaluation and comparison of fault diagnosis performance for the PD and MSPM approaches. In particular, previous sections detailed how we set up case studies and provided a short overview of each approach to refresh the reader. Finally, this section presented details on remaining aspects of the testing methodology. Together, all of this information describes the typical methods used to obtain the various results presented and analysed in the next chapter.

Chapter 6

Results and Discussion

This chapter considers the fault diagnosis performance of the PD approach in detail. First, our implementation of the PD approach with inference engine enhancements is verified to ensure that observation of anomalous results are not due to implementation errors. Next, fault diagnosis performance results for the PD approach on the two-tank system are presented and discussed. Following this detailed analysis, overall fault diagnosis performance for the PD approach on the two-tank system is summarised and presented alongside a summary of overall fault diagnosis performance for the MSPM approach on the same system. Following a comparative evaluation of the performance of the PD and the MSPM approaches, the PD approach is further investigated for scalability. The chapter then concludes with some caveats and a discussion of the limitations and some future directions of research for the PD approach.

6.1 Verifying implementation of the PD approach

The intention of this section is to verify the implementation of the PD approach using the two-tank system described in Section 5.2.1, but with a simplified DBN model. The ability of the inference engine presented in Section 3.2 to discriminate between normal and abnormal process behaviour is considered first. Next, the solutions described in Section 5.3.3 for dealing with numerical issues were implemented and tested. Subsequently, the computational enhancements for the inference engine, presented in Section 3.3, were implemented and tested. The section concludes with a summary of the findings.

6.1.1 Case study and testing methodology

The case study considered in this section is the sN configuration of the two-tank system. Only the aConn fault is tested in this case and this is the only fault modelled in the DBN. Therefore, the DBN only models two possible operating modes: normal and abnormal operation caused by the aConn fault. As a result, there is no need to further consider root cause analysis since the detection of fault conditions will imply

identification of the root cause. Also note that, unless otherwise stated, the same testing methodology described in Chapter 5 applies to this subsection.

Hyper-parameters for the PD approach were chosen to be $c = 0$ and $K = \infty$. This ensures that no transitions are combined with one another when creating hypotheses and, thus limits the degree to which combining transitions may affect observed results. Also note that the values of these parameters are acceptable for the particular DBN model being used since a maximum of only four transitions can be enumerated from an approximate belief state. Therefore, there is little need to manage the amount of transitions being enumerated from approximate belief states as well as the number of hypotheses that may be created from these transitions – both values may only be a maximum of four.

Furthermore, note that the hyper-parameters Pr_T and $T_{\max}$ only apply to the transition enumeration enhancement and are not used when the enhancement is not enabled.

6.1.2 Results for base case inference engine

Using the inference engine directly as presented in Section 3.2, the aConn fault was easily diagnosed. Plotting the ALI values reported by the PD approach over time as shown in Figure 6.1 reveals that the PD approach was easily able to discriminate between normal and abnormal process behaviour for this case.

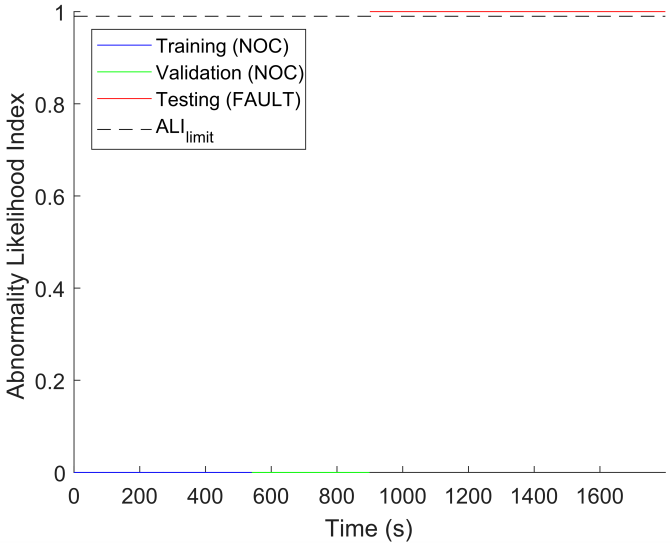


Figure 6.1: PD approach using base case inference engine: Abnormality likelihood index monitoring chart.

Processing of all 1800 samples was completed in 6 minutes and 26 seconds, averaging 0.2145 seconds processing time per sample.

6.1.3 Enabling solutions for numerical issues

Some challenges encountered during the implementation of the PD approach were highlighted in Section 5.3.3. Among these challenges is a particular type of numerical issue which affected the covariance matrices of Gaussian probability distributions. Solutions to avoid these issues were presented in the same section. These were subsequently incorporated into the base case inference engine and tested using the same case study and hyper-parameters.

Fault detection performance was found to be no different from that in the base case, suggesting that the added implementation of the solutions does not degrade performance of the base case inference engine. However, a slight increase in average sample processing time was observed compared to the base case inference engine. In this case average sample processing time was observed to be 0.2234 seconds, adding 16 seconds to the total processing time for the base case. Note that all subsequent tests have these solutions enabled, this stabilises the base case inference engine.

6.1.4 Enabling transition enumeration

Recall from Section 3.3.1 that transition enumeration facilitates a more practical approach to performing the inference calculations which produce a new approximate belief state. The enhancement does this by choosing which transitions to enumerate instead of enumerating all transitions from a particular approximate belief state. This reduces sample processing times and becomes increasingly beneficial for DBNs which feature a large number of operating modes. However, it is important to be aware that inadequate choices for Pr_T and $T_{\max}$ may result in poor performance.

Setting Pr_T to 1 and $T_{\max}$ to 2, limits the maximum number of transitions that can be enumerated from an approximate belief state to always be two. Since only four transitions can ever be enumerated from an approximate belief state for this case, these settings severely limited the ability of the PD approach to investigate the likelihoods of other transitions which were not enumerated. As a result, the normal to normal transition was always enumerated alongside only one other transition. Plotting ALI over time for this case, as shown in Figure 6.2, reveals that the fault could not reliably be detected using these settings. The results were observed due to the lack of enumeration of the fault-to-fault transitions.

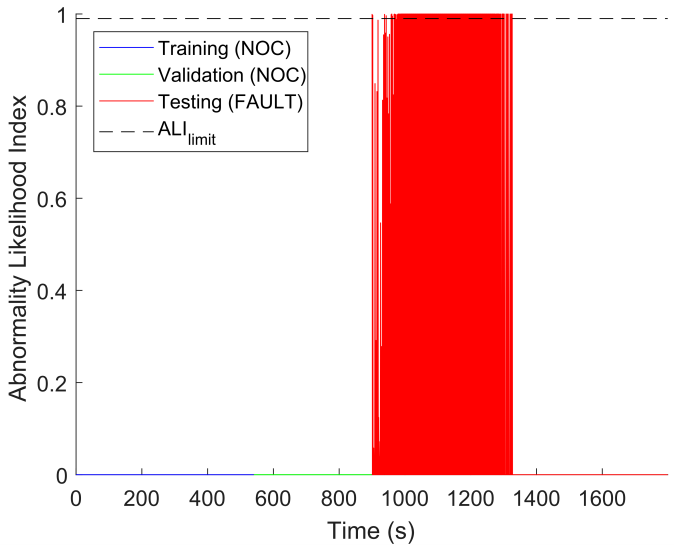


Figure 6.2: PD approach using enhanced inference engine (transition enumeration with inadequate $\Pr_T$ and $T_{\max}$): Abnormality likelihood index monitoring chart.

For this case, with $\Pr_T = 1$ and $T_{\max} = 2$, it was observed that transition enumeration actually worsened fault diagnosis performance of the PD approach. This insight reveals an important caveat for the PD approach with respect to the transition enumeration enhancement. The implication thereof is that the user must ensure that a sufficient number of transitions can be enumerated for the purposes of diagnosing any particular fault or set of faults in the case of multiple simultaneous faults. One approach to choosing values for $\Pr_T$ and $T_{\max}$ is to first set $T_{\max}$ according to the capabilities of the computer on which the PD approach will run and then choose $\Pr_T$ as desired, keeping in mind the *a priori* likelihood for fault conditions.

Following this approach, it is known that $T_{\max}$ cannot exceed four in our case, therefore choosing $T_{\max} \geq 4$ is sufficient for the first part. The second part depends on the *a priori* likelihood for fault conditions since $\Pr_T$ operates on the *a priori* likelihoods for transitions. This means that, in addition to the normal to normal transition, at least those transitions describing normal to fault component status changes should be enumerated from every approximate belief state under NOC. The *a priori* likelihood for fault conditions was discussed in Section 3.4.3. Using a value of $\alpha_{\text{nf}} = 0.001$ (noting that $m_{\text{opmodes}} = 2$), means that $\Pr_T$ should be at least $1 - \frac{\alpha_{\text{nf}}}{m_{\text{opmodes}}} = 0.9995$ to achieve the desired results.[1] Choosing $\Pr_T = 0.9999$ shows, in part, the benefits of using the transition enumeration enhancement for this case.

These settings have an interesting effect on the number of transitions typically enumerated from an approximate belief state under NOC. In particular, this number

[1]Note that 0.9995 is the absolute minimum recommended value for $\Pr_T$ and may still produce poor results due to it being on the threshold below which the values for $\Pr_T$ will be inadequate.

was always four transitions when using the base case inference engine, whereas with transition enumeration, at $\Pr_T = 0.9999$ and $T_{\max} = \infty$, this number was typically found to be two transitions. After manifestation of the fault the number of transitions enumerated using the base case engine remained four, whereas that number was found to typically be three for the implementation with transition enumeration. This gives evidence that reasonable settings for transition enumeration allows similar fault diagnosis with reduced sample processing times: the ALI plot over time was essentially the same as in Figure 6.1. On the other hand, the average sample processing time was found to be 0.1312 for the implementation with transition enumeration – a 41.3% improvement over the stabilised base case inference engine.

6.1.5 Enabling subsystem decomposition

Recall from Section 3.3.2 that subsystem decomposition aims to facilitate better management of hypotheses, allowing the PD approach to effectively maintain more hypotheses for, typically, the same amount of memory when compared to not using subsystem decomposition. Subsystems were specified in the same manner as described in Section 5.3.2 for the case study investigated here. Note that the second subsystem (tank) is the only subsystem containing a discrete RV, i.e. the status of the connecting pipe. Also note that this does not negatively affect the ability of the first subsystem to maintain hypotheses, but only means that the first subsystem only contributes to approximating the probability distributions of the continuous RVs associated with the first tank.

In this subsection, subsystem decomposition was implemented on the stabilised base case inference engine without the transition enumeration enhancement, i.e. for comparison with and without subsystem decomposition. This meant that each subsystem would maintain four hypotheses at all times since one hypothesis would be created from each transition for each subsystem. Similar fault diagnosis performance was observed in both cases, i.e. with and without subsystem decomposition. However, when using subsystem decomposition,average sample processing time increased due to the added computations needed to maintain double the number of hypotheses: sample processing times in this case was found to be 0.2334 seconds – a 4.5% degradation when compared with the stabilised base case inference engine.

Combining both transition enumeration ($\Pr_T = 0.9999$ and $T_{\max} = \infty$) and subsystem decomposition also had no negative impact on fault diagnosis performance when compared with results for the base case inference engine. Furthermore, average sample processing time in this case was found to be 0.1402 seconds – a 37.2% improvement over the stabilised base case inference engine.

6.1.6 Remarks

In this section it was observed that our implementation of the PD approach seems to function as intended. Testing the PD approach using the base case version of the inference engine revealed that the PD approach was easily able to discriminate between normal and abnormal process behaviour. The result was found to be the same when solutions to avoid numerical issues were added to the base case version of the inference engine, and this also held to the enhanced, i.e. with transition enumeration and subsystem decomposition, versions of that inference engine.

Furthermore, we noted that enumeration had the largest effect on average sample processing times. When compared with the stabilised base case inference engine, enabling transition enumeration reduced average sample processing time by 41.3%, and further enabling transition enumeration as well as subsystem decomposition saw this change to a 37.2% reduction in average sample processing time. It was also noted that a particular caveat applies to the use of transition enumeration: unsuitable settings for the hyper-parameters Pr_T and $T_{\max}$ could nullify any gains in sample processing times by making the PD approach unable to discriminate between normal and abnormal process behaviour as a result of enumerating too few transitions from any particular approximate belief state.

6.2 Testing the PD approach on the two-tank system

The two-tank system (in Section 5.2.1) features different configurations of varying complexity. Under each configuration, a number of different faults may manifest. In Section 3.4 and Section 5.3.2, the approach used to create the DBN models used in our approach was broadly presented. To better contextualise the fault diagnosis performance results presented in this section, some attention is first given to the specific nature of DBN models for the two-tank system.

Firstly note that the sN and N configurations share the same DBN design but we use different CPD parameters because of the introduction of process noise in the N configuration. Secondly, the NC and NCR configurations each use a different DBN design due to regulatory control in the NC configuration and regulatory control with recycle in the NCR configuration. Key differences between the DBN designs only affect continuous RVs and the edges between those RVs. Thus, the number of operating modes and therefore the total number of possible transitions that can be enumerated from an approximate belief state are the same for all two-tank DBNs.

All sensors (i.e. flows and pressures) were modelled as binary status components

and could be either functioning normally or be faulty.[2] There were three of these per tank. Additionally, the pipe connecting the two tanks was modelled as a binary status component as well. This meant that each of the DBNs for the two-tank system modelled a total of 128 operating modes with 128^2 (or 16384) possible transitions from an approximate belief state. Even though the two-tank system is the smallest case study investigated in this research, the number of operating modes modelled by the two-tank system's DBNs make this system challenging fault diagnosis using the PD approach. Fortunately the enhancements for the inference engine presented in Section 3.3 provide better management of computational resources in order to make sample processing more efficient.

The rest of this section summarises the observed fault diagnosis performance of the PD approach for the various configurations of the two-tank system. The raw data for this section as well as plots of the simulated data are made available in Appendix D and Appendix C, respectively.

6.2.1 Configuration sN

The sN configuration features only sensor noise, no regulatory control, and no recycle. Under this configuration, all faults were easily noticeable on the data plots shown in Appendix C.1, Figures C.1 through C.4. This was observed due to the vast difference in the magnitude of sensor noise compared to gross change in the values of observed PVs. It should be noted, however, that the incipient nature of the iConn fault caused affected PVs to slowly drift away from their nominal values in contrast to all other faults – all of which were abrupt and immediately caused large changes in the observed values of affected PVs.

Fault detection:
Unsurprisingly, the PD approach was able to detect all four faults under the sN configuration as summarised in Table 6.1. AUC was found to be relatively high in all cases, indicating good ability of the PD approach to discriminate between normal and abnormal process behaviour for this configuration. F1 scores were similarly high across all faults, indicating low FAR and TAR under this configuration. Detection delay results, however, show that the iConn fault was the only fault not detected immediately after manifestation.

[2]Note that although this research focused on sensor bias faults, the use of bias terms in the implementation of the faulty status (see Section 3.4.1.3) also enables diagnosis of sensor drift faults.

Table 6.1: Two-tanks (sN): Fault detection performance of PD approach.

Metric\Fault	aConn	iConn	P1s	P2s
AUC	1.000	0.936	1.000	1.000
F1 score	1.000	0.908	1.000	1.000
Detection delay (#samples)	0	168	0	0
Sample processing time (sec)	16.885	17.905	15.189	15.517

Due to the nature of the iConn fault, detection immediately after manifestation of the fault is challenging. This happens because the PD approach only has available a history of NOC process behaviour at the time when the fault manifests and since the first samples observed after manifestation of the fault are similar to NOC, no alarms are reported. In fact, the PD approach only detects the fault after processing 168 samples. Reducing the ALI limit may reduce detection delay by making the PD approach more sensitive at the cost of potential increase in FAR. However, even reducing the ALI limit to as low as 0.0001 to avoid increased FAR only lowers detection delay to 125 samples in this case.

Therefore, an alternative approach to reducing detection delay for this fault is desirable. In their research, Lerner *et al.* (2000) and Lerner (2002) use a smoothing technique which allows their fault diagnosis approach to use "future" process behaviour to influence diagnosis results. Although such a smoothing technique may benefit the diagnosis of incipient faults for our approach, it was not implemented due to the increased computational burden: the average sample processing times for the system were already high (see Table 6.1) and far exceeding the requirement for real-time fault diagnosis, so a multiplicative increase in these times for smoothing is not practical.

The differences in average sample processing times between the data sets can be attributed to the different behaviour of the PD approach after manifestation of a fault. These differences in behaviour of the PD approach are caused by the recursive nature of transition enumeration, hypothesis creation, and belief state approximation. Therefore, average sample processing times vary depending on the number of transitions enumerated and the number of hypotheses maintained per subsystem. Also note that since all the NOC data is the same for all four data sets under this configuration, differences in average sample processing time is only due to behaviour of the PD approach after manifestation of a fault.

Root cause analysis:

Analysing the component ALI plots for each fault, it was found that the correct root cause was highlighted (almost) immediately for three of the four faults. In all three of these cases, the root cause was the only fault cause to be highlighted — showcasing exceptional isolability for these faults. The exception, as expected, was found to be the iConn fault which did not have its root cause highlighted at all, but instead had two spurious fault causes highlighted. These results are summarised in Table 6.2.

Table 6.2: Two-tanks (sN): Root cause analysis performance of PD approach.

Metric\Fault	aConn	iConn	P1s	P2s
Root cause highlighted	Yes	No	Yes	Yes
Root cause position	1st	N/A	1st	1st
Number spurious causes	0	2	0	0

It is worth noting that since the PD approach can incorporate observations/evidence about discrete RVs as well, the correct root cause for the iConn fault could in practice be obtained using a troubleshooting approach: if a user were to investigate the fault cause initially with the highest ALI, i.e. the sensor for P_1 (see Figure D.2), and determine that it is functioning normally, then the PD approach can incorporate this evidence into its future analysis. Doing this highlights different fault causes which unfortunately do not include the root cause yet. However, investigating the new fault cause with the highest ALI, namely the sensor P_2, and determining it to be functioning normally produces root cause analysis results similar to that observed for the other faults under the sN configuration. Employing this troubleshooting method forces the PD approach to spend more time investigating alternative fault causes as well as different estimates for bias terms. Doing this repeatedly for the iConn fault in this case, eventually enabled the PD approach to better track the slow change in the constriction resistance caused by the fault.

6.2.2 Configuration N

The N configuration features process and sensor noise, no regulatory control, and no recycle. The addition of process noise in this case makes the N configuration more realistic than the sN configuration without introducing large amounts of complexity.

Fault diagnosis results observed under the N configuration were very similar to fault diagnosis results observed under the sN configuration. In fact, only a minor difference was observed for the root cause analysis of the aConn fault where the sensor F_{12} was slightly highlighted as a spurious fault cause (see Figure D.5). This was observed due to process noise increasing the variance of the PV F_{12}, causing the PD approach to consider the possibility of a slight bias in the sensor F_{12} after manifestation of the fault. Despite this, the correct root cause was highlighted and process behaviour was tracked correctly after manifestation of the fault.

Fault diagnosis performance is summarised in Table 6.3 for fault detection and Table 6.4 for root cause analysis. Note that the PD approach exhibited the same behaviour for the iConn fault under this configuration as it did under the sN configuration. Due to the relative similarity of the two configurations, the same analysis of results observed for the iConn fault under the sN configuration applies here.

Table 6.3: Two-tanks (N): Fault detection performance of PD approach.

Metric\Fault	aConn	iConn	P1s	P2s
AUC	1.000	0.935	1.000	1.000
F1 score	1.000	0.906	1.000	1.000
Detection delay (#samples)	0	191	0	0
Sample processing time (sec)	17.926	18.761	15.892	16.122

Table 6.4: Two-tanks (N): Root cause analysis performance of PD approach.

Metric\Fault	aConn	iConn	P1s	P2s
Root cause highlighted	Yes	No	Yes	Yes
Root cause position	1st	N/A	1st	1st
Number spurious causes	1	2	0	0

6.2.3 Configuration NC

The NC configuration is significantly more complex than the N configuration due to the introduction of regulatory control. In this case, the control system creates cyclic relationships between PVs in the underlying system, making it more difficult to identify the root cause of fault conditions when a fault is detected. The control system does this by introducing inverse relationships between P_1 and F_1, and P_2 and F_2. The data plots in Appendix C.3 show that the process was in a stationary state under NOC before manifestation of any fault. Once a fault manifests, the control system attempts to reduce the effects of the fault on the controlled PVs, i.e. P_1 and P_2, by adjusting the manipulated PVs, i.e. F_1 and F_2. The action of the control system in this case essentially makes detection of abrupt faults easier, due to gross changes in the observed values of multiple PVs. However, this also causes the effects of a fault to show strong symptoms in measured PVs further away from the root cause, making it more difficult to identify the root cause of fault conditions by inspection of the data.

Fault detection:
The PD approach was able to detect all faults as summarised in Table 6.5. Similar fault detection performance results to those for the sN and N configuration were observed: high AUC values, and high F1 scores, only the iConn fault featuring detection delay, and high sample processing times.

Compared to the sN and N configuration a significantly higher detection delay for the iConn fault was observed, i.e. 271 samples under NC versus 191 and 168 samples under N and sN respectively. This result was observed due to the joint nature of the fault and the actions of the control system. Since the iConn had a slow drift effect on F_{12}, the control system was able to efficiently counter the effects of the fault on P_1 and P_2 by slowly adjusting F_1 and F_2. This made it more difficult to detect the

iConn fault early on — the PD approach still inferred normal process behaviour the entire time until abruptly detecting fault conditions (see the plot of ALI over time in Figure D.10).

It is also worth noting the higher sample processing times observed for the data sets containing the P1s and P2s faults. In both cases, the PD approach inferred possible abnormal behaviour in two sensors (see Figure D.11 for the P1s fault and Figure D.12 for the P2s fault). Because these are marginal probabilities for abnormal behaviour in these sensors, this meant that the PD approach reported simultaneous abnormal behaviour in both components. This was not the case in reality since only one sensor was faulty in either case. However, in both cases the two faulty sensors were one for pressure and the other flow, and both sensors could be associated with a single control loop each time. This implies that the PD approach was diagnosing an issue with the control system and identifying the abnormal control loop in both cases. Although this behaviour might be convenient for isolating control system faults to particular loops, it would still be better if only the root cause is inferred to be abnormal. This would not only benefit root cause analysis, but would also reduce sample processing times since the PD approach would need to enumerate less transitions in such a case.

Table 6.5: Two-tanks (NC): Fault detection performance of PD approach.

Metric\Fault	aConn	iConn	P1s	P2s
AUC	1.000	0.860	1.000	1.000
F1 score	1.000	0.823	1.000	1.000
Detection delay (#samples)	0	271	0	0
Sample processing time (sec)	13.155	13.026	15.395	17.475

Root cause analysis:

Root cause analysis performance of the PD approach under the NC configuration was found to be excellent overall as summarised in Table 6.6. However, it should be noted that, with respect to the spurious causes listed, those for P1s and P2s were discussed above, while that for aConn was identified as due to process noise in the N configuration.

Table 6.6: Two-tanks (NC): Root cause analysis performance of PD approach.

Metric\Fault	aConn	iConn	P1s	P2s
Root cause highlighted	Yes	Yes	Yes	Yes
Root cause position	1^{st}	1^{st}	1^{st}	1^{st}
Number spurious causes	1	0	1	1

6.2.4 Configuration NCR

The NCR configuration introduces recycle to the NC configuration, further increasing complexity over the NC configuration. Note that the amount of water recycled back to the first tank is dependent on the outflow from the second tank – a PV manipulated by the control system. Therefore, the effect of recycle on process behaviour only plays a large role when an aggressive change in F_2 is caused by the control system.[3] Similar process behaviour was observed for the data sets containing the aConn, iConn, and P1s faults since these data sets featured no aggressive change in F_2 over time. In contrast, the data set containing the P2s fault featured aggressive change in F_2 immediately after manifestation of the fault which caused the effect of recycle to play a larger role in observed process behaviour.

In light of these observations, fault diagnosis performance results for the aConn, iConn, and P1s faults under the NCR configuration were expected to be similar to, if not the same as, those observed under the NC configuration. This expectation was met for fault detection as shown in Table 6.7, and partly met for root cause analysis as shown in Table 6.8. Naturally, the same explanations for most artefacts from the NC configuration apply here as well. One exception is that of poor root cause analysis for the iConn fault, which could be diagnosed using the troubleshooting approach described under the sN configuration.[4]

Furthermore, it was found that the PD approach was fully capable of handling the larger effect of recycle observed during the P2s fault. Fault diagnosis performance for the fault was found to to be very similar to that under the NC configuration. Once again, high ALI values were reported for both P_2 and F_2 sensors in this case. This implies that although the PD approach is capable of handling the effect of recycle, it is not capable of decoupling associated PVs in control loops.

Table 6.7: Two-tanks (NCR): Fault detection performance of PD approach.

Metric\Fault	aConn	iConn	P1s	P2s
AUC	1.000	0.858	1.000	1.000
F1 score	1.000	0.822	1.000	1.000
Detection delay (#samples)	0	272	0	0
Sample processing time (sec)	14.451	15.748	16.287	19.527

[3] This can be seen when comparing the data plots in Appendix C.4 to those in Appendix C.3, i.e. NCR configuration versus NC configuration.

[4] It should be noted that root cause analysis for the iConn fault worked flawlessly in past revisions. However, we were unable to note the cause for different results and document this update to reflect the most recent fault diagnosis performance of the PD approach.

Table 6.8: Two-tanks (NCR): Root cause analysis performance of PD approach.

Metric\Fault	aConn	iConn	P1s	P2s
Root cause highlighted	Yes	No	Yes	Yes
Root cause position	1st	N/A	1st	1st
Number spurious causes	1	2	1	1

6.2.5 Effects of hyper-parameter selection

The effects of different hyper-parameter values for c and K was investigated and found to have no negative effect on fault diagnosis performance over a selection of values. However, varying c and K did have an effect on sample processing times as a result of this altering the number of hypotheses maintained and the number of transitions enumerated from approximate belief states. This investigation was done using the aConn fault data set under the N configuration of the two-tank system. Sample processing times are reported in Table 6.9.

Table 6.9: Effect of hyper-parameters c and K on sample processing time (seconds) using the aConn fault data set under the N configuration of the two-tank system.

$c\backslash K$	1	5	100
0	4.65	8.45	22.94
0.0001	4.61	8.38	23.28
1	4.82	8.61	23.61
10000	16.54	18.59	18.72
∞	18.76	18.59	18.84

Supplementing this table, Table 6.10 and Table 6.11 show the median number of hypotheses maintained per subsystem and median number of transitions enumerated, respectively.[5]

Typically c and K, along with $\Pr_T$ and $T_{\max}$, control these values by intelligently discarding improbable process behaviour trajectories. In this case, increasing c causes the PD approach to combine more dissimilar transitions with one another to create hypotheses which contain all of the information of those transitions. Using these hypotheses to then approximate a belief state results in an increased number of transitions needing to be enumerated due to information from less likely transitions being maintained. This, in turn, causes the observed overall increase in sample processing time as c increases. Similarly, the observed overall increase in sample processing time as K increases is caused by the same effect, albeit more directly – i.e. increasing K results in more hypotheses being maintained per subsystem, including less likely and relatively unlikely ones for high values of K.

[5]Note that median values were used in order to limit the effect of outliers.

Table 6.10: Effect of hyper-parameters c and K on the median number of hypotheses maintained per subsystem using the aConn fault data set under the N configuration of the two-tank system. Note maximum number of hypotheses maintained per subsystem is $K + 1$ (see Section 5.3.3).

$c \backslash K$	1	5	100
0	2	6	101
0.0001	2	6	101
1	2	6	101
10000	2	3	3
∞	2	2	2

Table 6.11: Effect of hyper-parameters c and K on the median number of transitions enumerated using the aConn fault data set under the N configuration of the two-tank system.

$c \backslash K$	1	5	100
0	44	78	177
0.0001	44	78	177
1	44	78	177
10000	122	177	177
∞	177	177	177

6.2.6 Remarks

This section illustrated that the PD approach is capable of diagnosing an array of faults on multiple different configurations of the two-tank system. Typical fault detection and root cause analysis results were excellent for the PD approach. However, addressable issues with respect to the diagnosis of incipient faults and artefacts in diagnosis results under the NC and NCR configurations were observed.

Unfortunately, due to lack of information these findings cannot be directly compared with those of Lerner *et al.* (2000) and Lerner (2002). However, Lerner *et al.* (2000) did claim to reasonably track process behaviour under incipient fault conditions. Therefore, it was noted that implementation of smoothing, included in the systems of Lerner *et al.* (2000) and Lerner (2002), could benefit the PD approach by reducing detection delay for incipient faults. However, since poor sample processing times were observed for our implementation of the PD approach, this was not further considered.

Nonetheless, it was found that even without smoothing the PD was able to eventually detect the iConn fault under all configurations of the two-tank system after long detection delays. Furthermore, an addressable issue with respect to root cause analysis of the iConn fault under the sN, N, and NCR configurations was discussed. In

particular, the issue involved misidentification of the root cause when given only the process data. This was addressed by providing the PD approach with more information about the actual status of the misidentified components as might be obtained in a troubleshooting procedure on a real system. It was also noted that if the dynamics of the iConn fault were known, this issue would be avoided in root cause analysis. This result indicates that it may be worthwhile to incorporate some information about various fault dynamics into DBN models instead of naïvely modelling fault dynamics. This might be done by learning fault dynamics from existing, labelled data and incorporating this into the DBN model of a process.

The artefact in the fault diagnosis results for the P1s and P2s faults under the NC and NCR configurations was determined to be caused by the inability of the PD approach to decouple associated PVs in control loops. This artefact would cause the PD approach to identify both controlled and manipulated PV sensors as faulty when only the controlled PV sensor was faulty in reality. Nonetheless, the correct root cause was identified in these cases.

Finally, to conclude the performance analysis of the PD approach on the two-tank system, the effects of changing the hyper-parameters c and K on fault diagnosis performance was investigated. No negative effects on fault diagnosis performance were observed, however the different values of c and K did affect sample processing times. Unfortunately, the lowest sample processing times observed were still far too high to achieve real-time fault diagnosis using the PD approach.

6.3 Comparison of the PD and MSPM approaches for the two-tank system

The previous section showed that the PD approach was fully capable of diagnosing a suite of faults on the two-tank system under various different configurations. One of our key research questions is whether the PD approach warrants further investigation when compared to another readily implemented fault diagnosis approach, such as traditional MSPM. The MSPM approach was therefore also applied to the same two-tank system case study in order to make such a comparison. The same data sets in Appendix C were used in testing the MSPM approach and its raw fault diagnosis results are made available in Appendix E.

Note that fault diagnosis performance for both approaches is summarised on a per configuration basis in this section. This summary approach facilitates direct comparative evaluation of both fault diagnosis approaches for the different configurations, allowing for commentary on the impact of more complex configurations on fault diagnosis performance. Further note that this section does not make an attempt to provide detailed analysis of fault diagnosis performance for the

MSPM approach due to the abundance of research papers available on the topic (Venkatasubramanian *et al.*, 2003*c*). Instead, this section focuses on the fault detection and root cause analysis aspects of fault diagnosis for both PD and MSPM approaches and remarks on these findings.

6.3.1 Fault detection

Fault detection performance was summarised using four metrics, viz. AUC, F1 score, detection delay, and sample processing time. Results observed for the AUC, F1 score, and sample processing time metrics were averaged across all four faults under each configuration of the two-tank system. However, in all tests, non-zero detection delays were only reported for the iConn fault, therefore it was elected to compare detection delay metrics using only data for the iConn fault — i.e. a worst case scenario. All of this information is presented in Table 6.12 and is analysed in the text that follows.

Table 6.12: PD approach versus MSPM approach for two-tank system: Fault detection performance. Note all metrics, except detection delay, represent average values for a suite of four faults. Values for detection delay only represent the incipient iConn fault.

Configuration	**sN**		**N**		**NC**		**NCR**	
Metric\Approach	PD	MSPM	PD	MSPM	PD	MSPM	PD	MSPM
AUC	0.984	0.987	0.984	0.949	0.965	0.953	0.964	0.947
F1 score	0.977	0.991	0.976	0.976	0.956	0.979	0.955	0.976
Detection delay (#samples)	168	52	191	68	271	49	272	49
Sample processing time (sec)	16.374	$9.41e{-6}$	17.175	$8.43e{-6}$	14.763	$5.18e{-6}$	16.503	$2.50e{-6}$

Overall we observe similar fault detection performance for both approaches. However, the PD approach takes considerably longer to detect the incipient iConn fault, and sample processing time is orders of magnitude slower for the PD approach. We discuss these findings in more detail below, before considering how the PD approach stacks up against the MSPM approach for root cause analysis.

The average AUC values were similar for both approaches, both doing well at discriminating between normal and abnormal process behaviour for all configurations. However, it is also worth noting that the observed average AUC values decreased slightly overall when moving from less complex to more complex configurations. This phenomenon was also observed to be more noticeable for the MSPM approach when compared with the PD approach.

However, average F1 scores for the MSPM approach revealed that this phenomenon had little effect on maintaining low FAR and high TAR under all configurations. In contrast, the average F1 scores for the PD approach show the same trend as

the average AUC values for the PD approach when moving from less complex to more complex configurations. Findings in Section 6.2 reveal that the origin of this phenomenon for the PD approach can be traced back to high detection delays for the iConn fault. For the PD approach, under all configurations, lower AUC and lower F1 scores were only observed for the iConn fault — an explanation for the cause of these observation was also provided in Section 6.2.1.

Note that the impact of detection delays for the iConn fault on AUC values and F1 scores, when using the MSPM approach was found to be less noticeable. This was observed due to the manner in which detection limits were determined for the MSPM approach. Since detection limits for the MSPM approach were based on NOC training data, F1 scores for the MSPM approach were generally found to be lower than F1 scores for the PD approach under all configurations. Furthermore, compared with F1 scores for the PD approach, not as large a difference between the F1 scores for the abrupt faults and the F1 score for the iConn fault was observed when using the MSPM approach.

Comparing detection delays for the iConn fault, it can be seen that the PD approach performs significantly worse than the MSPM approach. Note that these long detection delays can potentially be addressed using a smoothing technique as described in Section 6.2.1. However, one would have to justify implementation of such a technique against additional sample processing time — recall that it was previously noted that our PD approach implementation is not yet suited to real-time fault detection because of high sample processing time.

In terms of sample processing time, the lightweight MSPM approach was found to be orders of magnitude faster than the PD approach. While processing a sample, the PD approach generally performs more computations than the MSPM approach, therefore the orders of magnitude differences were expected for this comparison. The key question, however, is whether or not these additional computations can yield better root cause analysis.

6.3.2 Root cause analysis

Overall root cause analysis performance across the different configurations of the two-tank system is summarised in Table 6.13 for both the PD approach and MSPM approach. In the table, three metrics are presented. The first metric describes the number of times the correct root cause was highlighted, on a relative contribution or component ALI plot, out of all faults tested under a given configuration. The second metric applies only to cases were the root cause was correctly highlighted, and describes the average position of the root cause, when ranking component ALI or relative contribution from highest to lowest, on a component ALI plot or relative contribution plot. Finally, the third metric describes the average number of spurious

fault causes indicated on a component ALI plot or relative contribution plot. Note that these averages were applied for all faults under a given configuration.

Table 6.13: PD approach versus MSPM approach on two-tanks system: Root cause analysis performance. Note all metrics represent average values for a suite of four faults.

Configuration	**sN**		**N**		**NC**		**NCR**	
Metric\Approach	PD	MSPM	PD	MSPM	PD	MSPM	PD	MSPM
Times root causes highlighted	3/4	4/4	3/4	4/4	4/4	4/4	3/4	4/4
Average root cause position	1	1	1	1	1	1.3	1	1.8
Average spurious causes	0.5	4.5	0.8	4.3	0.8	4.5	1.3	4.3

The first metric shows that the MSPM approach was able to highlight the correct root cause on each relative contribution plot whereas the PD approach could not do the same on each component ALI plot. This difference in results was caused by misdiagnosis of the iConn fault by the PD approach under the sN, N, and NCR configurations. As noted in Section 6.2.1 this was found to be an addressable issue and should, therefore, be viewed as relatively minor – especially in light of the performance of the PD approach for the other two metrics.

That being said, when the correct root cause was highlighted by the PD approach, it was found to be the highest ranked fault cause on all component ALI plots. In contrast, ranking of the root cause on contribution plots shifted lower down for the P1s fault as configuration complexity increased, causing an observed increase in average root cause position for the MSPM approach. Since the MSPM does not model effects of regulatory control and recycle, it simply produces higher relative contribution for the observed PVs which deviate the most from NOC. This leads to the MSPM approach highlighting all fault symptoms as possible fault causes and makes identification of the actual root cause of fault conditions a difficult task.

Unsurprisingly, Table 6.13 notes a high number of spurious fault causes highlighted on relative contribution plots under each configuration. In fact, since six PVs were observed, these spurious fault causes make up to 72% of the observed PVs. Therefore, even if the MSPM approach highlights the correct root cause of fault conditions on a relative contribution plot, one would still have to identify the actual root cause among a number of spurious fault causes. Accounting for this, it can be seen that the PD approach provides significantly better root cause analysis performance due it highlighting significantly lower numbers of spurious fault causes under all configurations.

6.3.3 Remarks

This section showed that the PD approach provides competitive fault detection performance when compared with the MSPM approach. However, the PD approach performed significantly worse than the MSPM approach in detection delay for the iConn fault as well as in terms of sample processing time. These areas can be addressed in future research to make fault detection performance of the PD approach more competitive with that of the MSPM approach. In particular, topics for future research with respect to improving diagnosis of the iConn fault are discussed in Section 6.2.6. On the other hand, reduction in sample processing time can be achieved through more optimal hyper-parameter selection (see Section 6.2.5) and through optimisation of the implementation (see Section 6.5.1).

It was later seen in this section that the PD approach provided significantly better isolation of root causes when compared with the MSPM approach. In particular, this was due to the observation that the PD approach generally produced better isolation of root causes. Furthermore, we highlight that the set of possible fault causes is not limited to observed PVs as is the case with MSPM. Rather with the PD approach, the user has more control over the set of possible fault causes as well as their granularity. For instance, one could specify multiple fault statuses for a pipe — which could include a blocked, burst, or punctured pipe.

Given the significant advantage that the PD approach has over the MSPM approach in terms of root cause isolation and control over the set of possible fault causes as well as their granularity, this warrants further investigation of the PD approach. This is further supported by Reis and Gins (2017) who remark that the fault diagnosis field has been saturated with good fault diagnosis strategies, and that current and future efforts should be directed toward to improving root cause diagnosis. The sentiment of the authors in this regard is to minimise process downtime while improving safe process operation.

6.4 Scaling the PD approach up to the five-tank system

In previous sections, the PD approach was tested on the two-tank system where it was found to provide good fault detection and root cause analysis performance, therefore warranting further investigation. Since the two-tank system is relatively small, testing the PD approach on a larger system such as the five-tank system became a primary focus for further investigation.

In this section only one fault was investigated under different configurations of the five-tank system. This fault would manifest as an abrupt partial blockage between the second and third tank and is similar to the aConn fault in the two-tank system.

Different DBN models were once again used for the different configurations of the five-tank system, and again all DBN models modelled the same set of process components. The same approach as in Section 6.2 was used to model these components, i.e. all sensors plus the connecting pipe which could have faults. However, after encountering memory usage issues it was decided to not model the sensor F_0 as a component.[6] In total the DBN models for the five tank system featured 2^{12} (4096) possible operating mode which could produce 2^{24} 16,777,216 possible transitions.

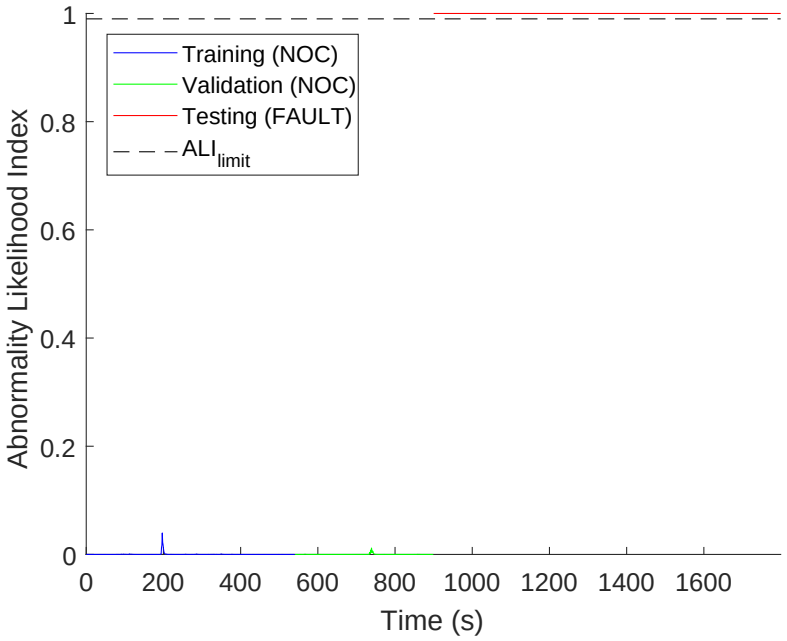
The significantly more complex DBN models would prove to be very challenging for the PD approach. This affected both design and modelling of the DBN models as well as their use in fault diagnosis. In this section, observations and issues with regard to scaling up the PD approach are highlighted and discussed. Simulation data is made available in Appendix F.

6.4.1 Configuration N

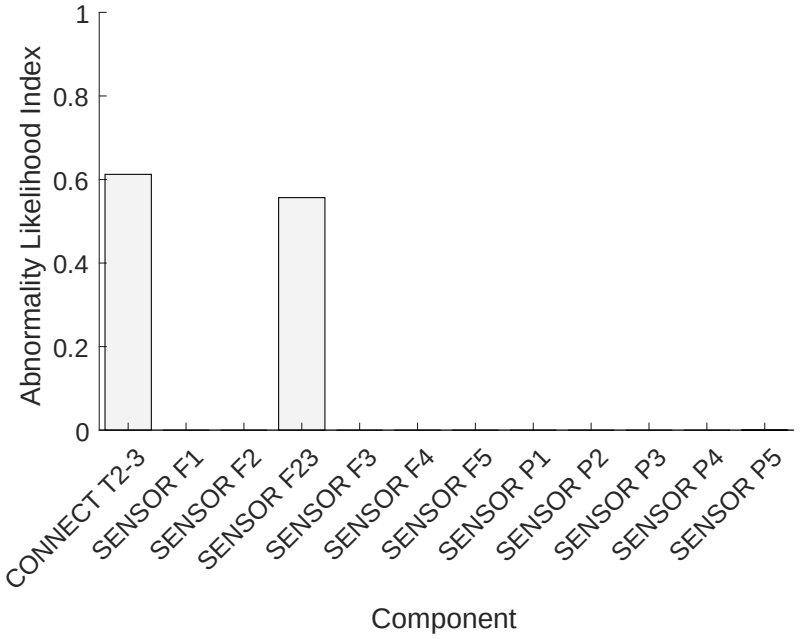
The N configuration for the five-tank, in similar fashion to that for two-tank system, features only process and sensor noise, no regulatory control and no recycle. Under this configuration the PD approach again produced excellent fault detection and reasonable root cause analysis results, as seen in Figure 6.3.

The spikes in ALI observed around 200 and 750 seconds were anomalies from the specific data set – repeated runs on alternative data did not exhibit such spikes. In this thesis, the CPD parameters for underlying PVs are determined from process data (see Section 3.4.3). Therefore if the process data does not sufficiently capture the variance of observed PV values, then this may cause a rapid increase and subsequent decrease in ALI over time to be observed due to an unexpected observation under NOC. Process data generated with a different random walk pattern and different seeds for sensor noise, yielded results with different or no spikes in ALI.

[6] Note that this, i.e. assuming that a component is extremely reliable, is a naïve solution to this problem. In Section 6.5.3 we remark on better solutions which may be explored in future research.



(a) Abnormality likelihood index over time.



(b) Average component abnormality likelihood indexes.

Figure 6.3: PD approach fault diagnosis results: Configuration N five-tank system with aConn type fault between tanks 2 and 3 manifesting after 900 seconds.

The component ALI plot highlighted the correct root cause and positioned the root cause higher than the other spurious fault cause in this case. However, it should be

noted that the highlighted spurious fault cause is related to the root cause and would, in fact, be identified as a root cause when using a traditional MSPM approach.

6.4.2 Configuration NC

The NC configuration adds regulatory control to the N configuration. For the five-tank system, this control system consists of five control loops each controlling one tank's pressure by manipulating that tank's outflow.

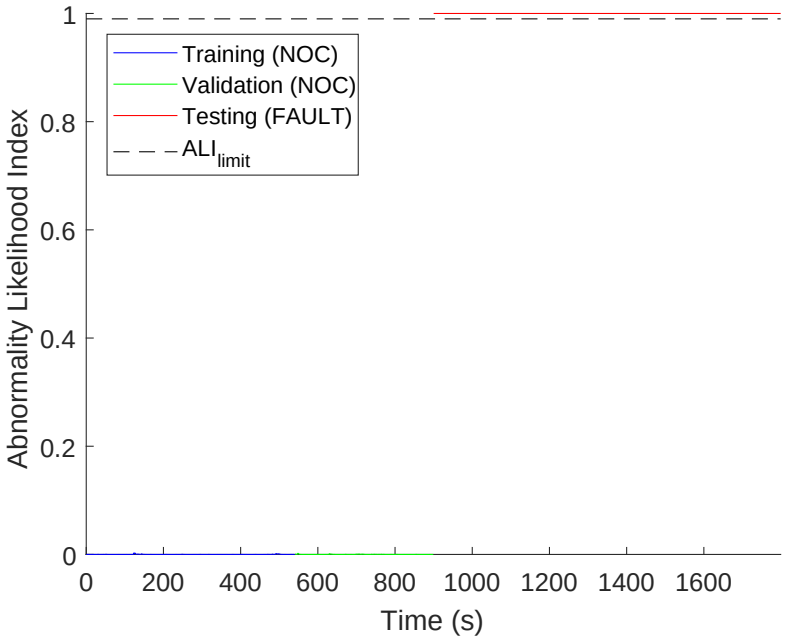
Unfortunately, the PD approach was unable to handle this configuration and would unreasonably escalate ALI values to unity for NOC data. In Appendix F it can be seen that the differences in the NOC data between the N and NC configurations were minimal. Despite this, the PD approach consistently reported high ALI values for NOC data, even for the same data used to define CPD parameters. Noting that artefacts were also observed in the fault diagnosis results from the two-tank system under configurations NC and NCR, it was therefore suspected that there may be some issue present in our implementation of regulatory control in the DBN model. The issue could not be resolved for the five-tank system in this work.

It should also be noted that this issue was further tested using similar three- and four-tank systems, and they were found to exhibit same behaviour. For the three- and four-tank systems under the N configuration, fault diagnosis results were similar to that observed under the N configuration of the five-tank system. Likewise, for the three- and four-tank systems under the NC configuration, ALI values would increase to unity for NOC data.

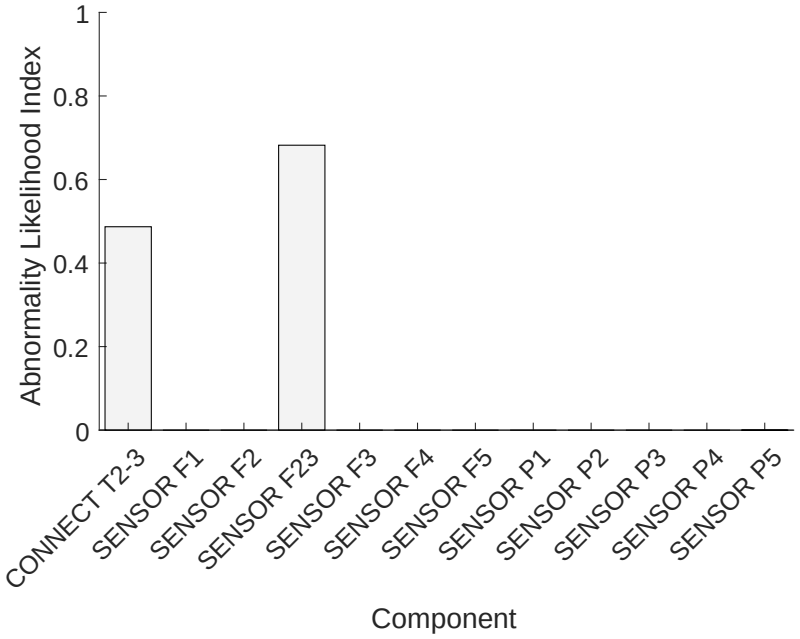
6.4.3 Configuration NR

The NR configuration adds recycle to the N configuration. Since regulatory control was observed to cause issues for the PD approach beyond two-tanks, it was elected to test the approach on a configuration with just process noise, sensor noise, and recycle, rather than NCR.

Despite abysmal performance of the PD approach for the NC configuration, the PD approach produced excellent fault detection and reasonable root cause analysis results, as shown in Figure 6.4, for the NR configuration. Once again, spikes in the ALI values were observed for the same reasons as previously conjectured in Section 6.4.1. Once again, using different process data with a different random walk pattern and different seeds for sensor noise, yielded results with different or no spikes in ALI. Root cause analysis results also highlighted the same two fault causes as was seen under the N configuration.



(a) Abnormality likelihood index over time.



(b) Average component abnormality likelihood indexes.

Figure 6.4: PD approach fault diagnosis results: Configuration NR five-tank system with aConn type fault between tanks 2 and 3 manifesting after 900 seconds.

6.4.4 Remarks

Scaling the PD approach up to five-tanks was computationally intensive as sample processing times averaged above 160 seconds. Nonetheless, the PD approach was capable of detecting and diagnosing the fault in both the N and NR configurations for the five-tank system. The similarities between the fault diagnosis results observed in the N and NR configurations strengthen the impression that the PD approach is capable of handling recycle and hence cyclic structures in processes. Furthermore, the failure of the PD approach to correctly classify NOC data as normal process behaviour in the NC configuration strengthens the impression that there are issues with our implementation of regulatory control in DBN models.

DBN design and modelling was found to be a difficult task throughout this research. This task can certainly be improved with future research by, for example, investigating different approaches to learning CPD parameters from data given the structure of a DBN, or learning the entire DBN model from data.

6.5 PD approach: Caveats, limitations, and future research

This chapter showed that the PD approach can capably handle fault diagnosis on the two-tank system. For this system, fault detection performance is competitive with the MSPM approach, albeit with a longer detection delay for the incipient fault and longer sample processing times. Scalability of the PD approach on the other hand, revealed that our implementation was unable to deal with regulatory control on systems featuring more than two-tanks. Nonetheless configurations without regulatory control, but including recycle, were capably handled by the PD approach.

These observations suggest that despite issues encountered with the PD approach, it performs well enough to warrant further investigation and future research. The intention of this section is to highlight avoidable pitfalls and as well as some aspects of the PD approach that could benefit most from further research. Therefore, this section focuses on important caveats, limitations, and some directions for future research with respect to the PD approach.

6.5.1 Profiling: Bottlenecks and potential speed-ups

Software profiling monitors the real-time execution of code and highlights the functions and lines of code which could benefit most from improvement, i.e. in terms of computational time and memory usage. This subsection discusses the results of profiling the PD approach in order to determine aspects of our implementation that need to be improved for the benefit of reducing sample processing time.

The sN configuration of the two-tank system was tested for profiling.[7] 900 data samples were generated with the aConn fault occurring after 600 seconds. The PD approach maintained the same DBN model as was used in Section 6.2.1. However, hyper-parameters were changed to $\Pr_T = 1 - 1e\text{-}12$, $T_{\max} = 10000$, $c = 1$, and $K = 5$ in light of the results observed in Section 6.2.5.

The profiler noted a time of 2 hours and 21 minutes to complete inference for all 900 samples, corresponding to 7.81 seconds per sample. Recalling that the sensor sampling rate was specified to be 1Hz, this sample processing time is inadequate for real-time fault diagnosis. Analysing the profiling results, however, it was found that most of the processing time is spent in one function and its children.

During inference for one time slice, this function (i.e. `generate_gaussians`) generates one Gaussian probability distribution for each transition, which had been marked for enumeration, using the DBN model. The profiler noted that 96.1% of the profiling time was spent on this function and its children. In particular, most of this time was spent on one child function in two separate function calls. This function, i.e. `enter_evidence`[8], computes a JPD for the DBN and was found to use 97.6% of the computational time used by `generate_gaussians`.

Naturally, since such a large portion of computational time was spent on this function and its children, reducing the time required by `generate_gaussians` and `enter_evidence` could massively reduce sample processing time.[9] The rest of this section discusses two approaches that can be used in future research to accomplishing this.

First, we discuss the requirement for computing a JPD twice in the same function. Given a transition to enumerate, the first computation uses the joint to determine the likelihood of observations (i.e. part of the normalising constant), and the second computation determines the probability distribution over the unobserved PVs at time slice $t + 1$ given those observations. Naturally, one would want to obtain both of these items by only computing a JPD once. However, the BNT used does not directly allow one to obtain the likelihood of the observations, hence this work-around was used.

Second, generating a Gaussian for each transition simply involves conditioning the DBN to only consider the set of particular component statuses specified by that transition. For example, the normal to normal transition requires that a probability distribution over the PVs be generated where all components are in, and maintain, the normal status. Considering this, it may be possible to generate these probability

[7] Note that only the computations of the inference engine were profiled.

[8] Note that the `enter_evidence` function is part of the BNT by Murphy (2002).

[9] It should also be noted that significant speed-ups may also be achieved through implementation in a different programming language – for example C++ which, per private communication with Prof. J.A. du Preez at Stellenbosch University, may be a factor of ten times faster compared to MATLAB® .

distributions in parallel rather than sequentially.

6.5.2 DBN modelling and learning from data

DBN modelling may be the most important aspect of the PD approach. When approximating a belief state for a process, the PD approach relies heavily on the predictions of process behaviour given a DBN model in order to explain process observations. Therefore, if predictions of process behaviour receive little support from process observations the PD approach may be unable to recover and thus misdiagnose process behaviour — similar to being stuck in a valley in optimisation problems.

Throughout the development of the PD approach for the two- and five-tank systems it was found that most anomalous results were caused by DBN modelling issues. For this thesis, DBN models were specified manually. This meant that human error would be a potential factor in the DBN modelling stage. Therefore, during the development process when the PD approach would report anomalous results, this would often be resolved by analysing the DBN model. Throughout this process better solutions to manage potential issues were also discovered, as highlighted in Section 5.3.3. However, the issues causing long detection delay for incipient faults and anomalous results for larger systems with regulatory control could not be resolved in the available time. It is hoped that alternative DBN modelling approaches may yield better performance in this regard.

One such approach may be to learn DBN models from historical process data. Although it is outside the scope of this research to investigate this, it should be noted that Koller and Friedman (2009) describe a number of approaches to learning different aspects of a DBN model. Of particular interest is that of learning CPD parameters given the structure of a DBN model. Adding this to the PD approach could allow a user to specify the structure of a DBN model from existing process knowledge, and then learn the parameters of CPDs from historical process data — at the very least under NOC. This may reduce modelling requirements for the PD approach as well as limit the extent to which human error may result in the specification of inadequate DBN models. Therefore, this should be investigated in future research.

Another topic of interest with respect to DBN modelling, is the incorporation of nodes to diagnose the manifestation of novel faults. An important limitation of the PD approach in this thesis is that it can only diagnose faults for a defined set of components. This means that if the PD approach is able to detect the manifestation of novel faults, its root cause analysis will be severely limited by the set of user-defined components. Therefore, a way to attribute the cause of fault conditions to a novel component should be investigated in future research.

It may then be possible to combine the two aforementioned topics for learning fault behaviour, and then add the learned fault behaviour to the DBN model. This could

facilitate learning the rest of a DBN model from the structure of that DBN model under NOC and historical process data containing labelled faults.

6.5.3 Scalability to larger processes

It is important to note that computational memory requirements and sample processing time are dependent on the number of operational modes (i.e. total number of combinations of component statuses). This is problematic for large processes which have many components to model. Consider the Tennessee Eastman process (Ricker, 1996) for which 21 faults are modelled. Without modelling any other possible faults in a DBN model for the PD approach, this process features a total of 2^{21} operational modes. This means that a total of 2^{42} transitions could be enumerated from each approximate belief state. Even with the transition enumeration enhancement, scaling up to processes as large as this seems impractical — especially when considering memory requirements. In this case, just to store the probability of each transition in a double precision array, MATLAB® would require approximately 36 TB of RAM.

A possible approach to tackling to this problem may be to use a two-stage DBN-based PD approach. In particular, design the approach such that the first stage isolates the cause of detected fault conditions to a subsystem, then the second stage could perform a more detailed root cause analysis within that subsystem. This approach would allow for streamlined DBN models in the first stage, as the PD approach would only have to determine the subsystem causing abnormal process behaviour. This should also benefit sample processing time under NOC due to significantly less transitions needing to be processed when compared with the current PD approach. Subsequently, once a fault is detected the second stage could run in parallel with the first stage, offering more detailed root cause analysis while the rest of the process is monitored continuously.

There also exists potential to combine such an approach with the ideas provided in the previous section. Doing this could create a highly adaptable fault diagnosis approach with excellent root cause analysis abilities.

As an alternative, one could exploit these root cause analysis abilities of the PD approach and use it as a dedicated tool for root cause analysis. Doing this would involve using a faster fault detection approach, such as MSPM, and then using the PD approach only to isolate fault causes once a fault is detected.

6.5.4 Implementation of variational inference methods

At the end of Chapter 3 (page 85) we mentioned that the inference engine used in the PD approach may benefit from more advanced probabilistic methods. In particular, we highlighted that the use of a variational inference method such as expectation

propagation (EP) may greatly reduce computational requirements. EP attempts to approximate a complex, intractable probability distribution p by a simple, tractable one q using an iterative method which minimises the Kullback-Leibler divergence from p to q (Minka, 2001). In our case, p is the exact belief about the process state and q is our approximate belief about the process state. Using this approach q can still be represented as a hybrid mixture of Gaussians and thus the same fault detection and root cause analysis methods described in Section 3.2.3 may be applied to q.

Some potential advantages of using EP for inference include:

1. Avoiding back and forth conversion between moment and canonical forms, thus avoiding the cause of numerical issues described in Section 5.3.3.
2. Integration of the process model DBN and belief network BN in a single probabilistic graphical model such as a cluster graph which can facilitate message passing techniques (Koller and Friedman, 2009).
3. The ability to extend upon a cluster graph implementation which would allow for easier and less computationally expensive implementation of smoothing techniques. This may resolve the issues had with the diagnosis of incipient faults as previously mentioned.
4. Each cluster may also be used to represent the approximate belief about an individual subsystem's state. This can allow for more advanced subsystem decomposition which may further reduce computational requirements for the PD approach.

It is therefore recommended that future research also consider the use of advanced probabilistic methods to further improve upon the inference engine presented in 3.

Chapter 7

Conclusion

Problem statement In efforts to reduce the impact of human error on the operation of chemical and mineral processing plants, reliable process monitoring solutions attempt to assist plant operators and engineers to detect and diagnose process faults before significant loss is incurred. An existing solution, the traditional MSPM approach, is able to reliably detect abnormal process behaviour, but struggles to unambiguously identify the root cause of the abnormal behaviour. It was identified that this is caused by a lack of incorporation of existing process knowledge into the framework of the MSPM approach.

Proposed solution and implementation It was proposed to investigate a different fault diagnosis approach which directly incorporates process knowledge into its framework. Lerner *et al.* (2000) and Lerner (2002) present such an approach, using probabilistic methods to infer process behaviour given a particular process model. This model is in the form of a DBN, and would contain various models which each describe particular process behaviour given information about the operational status of various process components. In particular, these DBN models were able to describe normal process behaviour in addition to highly specific abnormal process behaviour caused by, for instance, a sensor fault or a blocked pipe. Using optimised methods, the authors could then use a DBN model to make predictions about process behaviour and infer, given observation of actual process behaviour, which combination of component statuses best describe that observation. Therefore, solving the fault diagnosis problem could be reduced to performing inference in a DBN using this approach.

A PD approach based on Lerner *et al.* (2000) and Lerner (2002) was therefore implemented and investigated in this thesis. A survey of recent DBN-based PD approaches was also performed, and it was determined that relatively little research had been done on the topic. Furthermore, published results presenting fault diagnosis performance for DBN-based PD approaches were typically found to not be useful for meaningful comparison with a traditional MSPM approach. In this regard, this thesis aimed to investigate the usefulness of the PD approach in comparison to the MSPM approach, while providing useful fault diagnosis performance metrics to facilitate

comparison with other fault diagnosis approaches.

The PD approach tested in this research also extended upon Lerner *et al.* (2000) and Lerner (2002) by including models for regulatory control systems and recycle streams inspired by Yu and Rashid (2013). Additionally, from the same paper, the concept of abnormality likelihood index (ALI) was implemented in the PD approach. This enabled the PD approach to function more similarly to the MSPM approach, facilitating direct comparison.

Findings Both fault diagnosis approaches were tested on a simple two-tank system case study. The case study featured multiple different configurations of varying levels of complexity, namely: (sN) sensor noise only, no control, no recycle; (N) process and sensor noise, no control, no recycle; (NC) process and sensor noise, control, no recycle; and (NCR) process and sensor noise, control, and recycle. Four faults were tested under each configuration, namely two structural faults (one abrupt and the other incipient) and two sensor bias faults (one for each tank pressure sensor).

It was found that the PD approach could provide competitive fault detection compared to the MSPM approach. However, longer detection delay for the incipient fault under all configurations as well as multiple orders of magnitude slower sample processing time compared to the MSPM approach would count against the fault detection performance of the PD approach. On the other hand, it was found that the PD approach was performed better at isolating root causes when compared with the MSPM approach. Generally, the PD approach was able to better isolate the root cause for each fault even under more complex configurations. In contrast, the MSPM approach struggled to isolate the root cause for each fault, with root cause analysis getting worse under more complex configurations. We also note that the PD approach provides more control over the set of possible fault causes than MSPM does — this would allow one to identify specific root causes instead of only closely related fault symptoms.

These findings motivated further investigation of the PD approach in respect to scaling up to larger processes. The PD approach was thus tested on a five-tank system case study. This case study also featured different configurations of varying complexity, namely N, NC, and NR (N plus recycle) configurations. Only one abrupt structural fault was tested under each configuration.

It was found that the PD approach could capably detect and diagnose the fault under the N and NR configurations. Under the NC configuration, on the other hand, the PD approach could not discriminate between normal and abnormal process behaviour, incorrectly classifying NOC data as abnormal process behaviour. We conjecture that the cause for this is related to the modelling approach used for regulatory control in the DBN model. Because of this, the PD approach would poorly predict process

behaviour under normal operation, this prediction would then not be supported by observations which would then force the PD approach to infer abnormal process behaviour. Unfortunately, the issue could not be resolved in this thesis.

Recommendations In conclusion, we believe that the PD approach warrants further research, especially in the directions of reducing sample processing time, improving diagnosis of incipient faults, and improving our implementation of regulatory control in DBN models. Future research may then focus on lowering modelling requirements of the PD approach as well as improving its scalability to larger processes. In particular, DBN modelling could benefit from learning CPD parameters as well as fault models from historical process data. Furthermore, splitting the PD approach into two separate stages, one for fault detection and the other for root cause analysis, may improve scalability by reducing sample processing time for real-time fault detection while providing powerful root cause analysis abilities. Alternatively, using more advanced probabilistic methods in variational inference, such as expectation propagation, may produce a more robust and scalable inference engine.

Bibliography

Aldrich, C. and Auret, L. (2013). *Unsupervised Process Monitoring and Fault Diagnosis with Machine Learning Methods*, vol. 48 of *Advances in Computer Vision and Pattern Recognition*. Springer, London. ISBN 9781447151845.

Askarian, M., Zarghami, R., Jalali-Farahani, F. and Mostoufi, N. (2016). Fault diagnosis of chemical processes considering fault frequency via Bayesian network. *The Canadian Journal of Chemical Engineering*, vol. 94, no. 12, pp. 2315–2325. ISSN 00084034. Available at: http://dx.doi.org/10.1002/cjce.22603

Atoui, M.A., Verron, S. and Kobi, A. (2015). Fault detection and diagnosis in a Bayesian network classifier incorporating probabilistic boundary. *International Federation of Automatic Control (IFAC)-PapersOnLine*, vol. 48, no. 21, pp. 670–675. ISSN 24058963. Available at: http://dx.doi.org/10.1016/j.ifacol.2015.09.604

Barber, D. (2011). *Bayesian Reasoning and Machine Learning*. Cambridge University Press, Cambridge. ISBN 9780511804779. Available at: http://dx.doi.org/10.1017/CBO9780511804779

Bullemer, P.T. and Reising, D.V.C. (2015). Managing Human Reliability: An Abnormal Situation Management Historical Perspective. In: *Mary Kay O'Connor Process Safety Center 2015 International Symposium*. Abnormal Situation Management Consortium, College Station, TX.

Chen, J. and Patton, R.J. (1999). *Robust Model-Based Fault Diagnosis for Dynamic Systems*, vol. 3 of *The International Series on Asian Studies in Computer and Information Science*. Springer, Boston, MA. ISBN 9781461373445. Available at: http://dx.doi.org/10.1007/978-1-4615-5149-2

Cover, T.M. and Thomas, J.A. (2005). *Elements of Information Theory*. 2nd edn. John Wiley & Sons, Inc., Hoboken, NJ, USA. ISBN 9780471748823. Available at: http://dx.doi.org/10.1002/047174882X

Dai, Y., Wang, H., Khan, F. and Zhao, J. (2016). Abnormal situation management for smart chemical process operation. *Current Opinion in Chemical Engineering*, vol. 14, pp. 49–55. ISSN 22113398. Available at: http://dx.doi.org/10.1016/j.coche.2016.07.009

De Keyser, V. and Leonova, A.B. (2001). *Error Prevention and Well-Being at Work in Western Europe and Russia*. Springer, Dordrecht, Netherlands. ISBN 9780792371007. Available at: http://dx.doi.org/10.1007/978-94-010-0784-9

Dey, S. and Stori, J. (2005). A Bayesian network approach to root cause diagnosis of process variations. *International Journal of Machine Tools and Manufacture*, vol. 45, no. 1, pp. 75–91. ISSN 08906955.
Available at: http://dx.doi.org/10.1016/j.ijmachtools.2004.06.018

Dong, D. and McAvoy, T. (1996). Nonlinear principal component analysis — Based on principal curves and neural networks. *Computers & Chemical Engineering*, vol. 20, no. 1, pp. 65–78. ISSN 00981354.
Available at: http://dx.doi.org/10.1016/0098-1354(95)00003-K

Downs, J. and Vogel, E. (1993). A plant-wide industrial process control problem. *Computers & Chemical Engineering*, vol. 17, no. 3, pp. 245–255. ISSN 00981354.
Available at: http://dx.doi.org/10.1016/0098-1354(93)80018-I

Dynkin, E.B. (1965). Markov processes. In: *Markov Processes*, vol. 121/122 of *Die Grundlehren der Mathematischen Wissenschaften*, chap. 3, pp. 77–104. Springer, Berlin, Heidelberg. ISBN 9783662000335.
Available at: http://dx.doi.org/10.1007/978-3-662-00031-1_4

Eaton, M.L. (1983). *Multivariate Statistics: A vector space approach*. John Wiley & Sons, New York. ISBN 0471027766.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874. ISSN 01678655.
Available at: http://dx.doi.org/10.1016/j.patrec.2005.10.010

Gertler, J. (1988). *Fault Detection and Diagnosis in Engineering Systems*. Marcel Dekker, New York. ISBN 9780824794279.

Hartikainen, J., Solin, A. and Särkkä, S. (2011). *Optimal filtering with Kalman filters and smoothers*. Version 1.4. Toolbox for MATLAB®, Department of Biomedical Engineering and Computational Sciences, Aalto University School of Science, Greater Helsinki, Finland.

He, S., Wang, Z., Wang, Z., Gu, X. and Yan, Z. (2016). Fault detection and diagnosis of chiller using Bayesian network classifier with probabilistic boundary. *Applied Thermal Engineering*, vol. 107, pp. 37–47. ISSN 13594311.
Available at: http://dx.doi.org/10.1016/j.applthermaleng.2016.06.153

Higham, N.J. (1988). Computing a Nearest Symmetric Positive Semi-definite Matrix. *Linear Algebra and its Applications*, vol. 103, pp. 103–118. ISSN 00243795.
Available at: http://dx.doi.org/10.1016/0024-3795(88)90223-6

Himmelblau, D. (1978). *Fault Detection and Diagnosis in Chemical and Petrochemical Processes*. Elsevier, Amsterdam; New York. ISBN 9780444417473.

Isermann, R. (1984). Process fault detection based on modeling and estimation methods — A survey. *Automatica*, vol. 20, no. 4, pp. 387–404. ISSN 00051098. Available at: http://dx.doi.org/10.1016/0005-1098(84)90098-0

Isermann, R. (2006). *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*. Springer, Berlin, Heidelberg. ISBN 9783540241126. Available at: http://dx.doi.org/10.1007/3-540-30368-5

Jolliffe, I. (1986). *Principal Component Analysis*. Springer Series in Statistics. Springer, New York, NY. ISBN 9781475719062.

Julier, S. and Uhlmann, J. (2004). Unscented filtering and nonlinear estimation. In: *Proceedings of the IEEE*, vol. 92, pp. 401–422. Institute of Electrical and Electronics Engineers (IEEE). ISSN 00189219.

Kaushik, P. and Khanduja, D. (2009). Application of Six Sigma DMAIC methodology in thermal power plants: A case study. *Total Quality Management & Business Excellence*, vol. 20, no. 2, pp. 197–207. ISSN 1478-3363. Available at: http://dx.doi.org/10.1080/14783360802622995

Kelly, B. and Lees, F. (1986). The propagation of faults in process plants: 1. Modelling of fault propagation. *Reliability Engineering*, vol. 16, no. 1, pp. 3–38. ISSN 01438174. Available at: http://dx.doi.org/10.1016/0143-8174(86)90070-3

Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and techniques*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA. ISBN 9780262259842.

Koller, D. and Lerner, U.N. (2001). Sampling in Factored Dynamic Systems. In: *Sequential Monte Carlo Methods in Practice*, chap. 21, pp. 445–464. Springer, New York, NY. ISBN 9781475734379. Available at: http://dx.doi.org/10.1007/978-1-4757-3437-9_21

Kourti, T. and MacGregor, J.F. (1995). Process analysis, monitoring and diagnosis, using multivariate projection methods. *Chemometrics and Intelligent Laboratory Systems*, vol. 28, no. 1, pp. 3–21. ISSN 01697439.

Kullback, S. and Leibler, R.A. (1951). On Information and Sufficiency. *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86. ISSN 00034851.

Lauritzen, S. (1996). Models for Mixed Data. In: *Graphical Models*, vol. 17 of *Oxford Statistical Science Series*, chap. 6, pp. 158–163. Oxford University Press, New York, NY. ISBN 9780191591228.

Lawler, G.F. and Limic, V. (2010). *Random Walk: A Modern Introduction*, vol. 123 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge. ISBN 9780511750854.

Lees, F. (2012). *Lees' Loss Prevention in the Process Industries: Hazard identification, assessment and control*. 4th edn. Elsevier, Burlington, MA. ISBN 9780123971890. Available at: http://dx.doi.org/10.1016/C2009-0-24104-3

Leopold, T.A., Ratcheva, V. and Zahidi, S. (2016). *The Future of Jobs: Employment, skills and workforce strategy for the fourth industrial revolution*. World Economic Forum, Geneva, Switzerland.
Available at: http://reports.weforum.org/future-of-jobs-2016/

Lerner, U.N. (2002). *Hybrid Bayesian networks for reasoning about complex systems*. PhD thesis, Department of Computer Science, Stanford University, Stanford, CA 94305, USA.

Lerner, U.N., Parr, R., Koller, D. and Biswas, G. (2000). Bayesian fault detection and diagnosis in dynamic systems. In: *32nd Conference on Artificial Intelligence (AAAI-00)*, pp. 531–537. American Association for Artificial Intelligence (AAAI), Austin, TX.

Li, X., McKee, D., Horberry, T. and Powell, M. (2011). The control room operator: The forgotten element in mineral process control. *Minerals Engineering*, vol. 24, no. 8, pp. 894–902. ISSN 08926875.
Available at: http://dx.doi.org/10.1016/j.mineng.2011.04.001

Lindner, B.S. and Auret, L. (2015). Application of data-based process topology and feature extraction for fault diagnosis of an industrial platinum group metals concentrator plant. *International Federation of Automatic Control (IFAC)-PapersOnLine*, vol. 48, no. 17, pp. 102–107. ISSN 24058963.
Available at: http://dx.doi.org/10.1016/j.ifacol.2015.10.086

Marlin, T. (2000). *Process Control: Designing processes and control systems for dynamic performance*. 2nd edn. McGraw-Hill, Boston, MA. ISBN 9780070393622.

Maurya, R.M., Rengaswamy, R. and Venkatasubramanian, V. (2004). Application of signed digraphs-based analysis for fault diagnosis of chemical process flowsheets. *Engineering Applications of Artificial Intelligence*, vol. 17, no. 5, pp. 501–518. ISSN 09521976.
Available at: http://dx.doi.org/10.1016/j.engappai.2004.03.007

Miller, R.G. (1981). *Simultaneous Statistical Inference*. Springer Series in Statistics. Springer New York, New York, NY. ISBN 9781461381242.
Available at: http://dx.doi.org/10.1007/978-1-4613-8122-8

Minka, T. (2001). Expectation Propagation for approximate Bayesian inference. In: Breese, J.S. and Koller, D. (eds.), *UAI '01 Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pp. 362–369. Morgan Kaufmann Publishers Inc., San Francisco, CA.

Miskin, J.J. (2016). *Control performance assessment for a high pressure leaching process by means of fault database creation and simulation*. Masters Thesis, Department of Process Engineering, Stellenbosch University.

Murphy, K.P. (2002). *Dynamic Bayesian Networks: Representation, inference and learning*. PhD Thesis, Department of Computer Science, University of California, Berkeley.

Patton, R.J. and Chen, J. (1997). Observer-based fault detection and isolation: Robustness and applications. *Control Engineering Practice*, vol. 5, no. 5, pp. 671–682. ISSN 09670661.
Available at: http://dx.doi.org/10.1016/S0967-0661(97)00049-X

Pearl, J. (1985). Bayesian networks: a model of self-activated memory for evidential reasoning. In: *7th Annual Conference of the Cognitive Science Society*, pp. 329–334. Cognitive Science Society, Irvine, CA.

Perry, R.H. and Green, D.W. (2007). *Perry's Chemical Engineers' Handbook*. 8th edn. McGraw-Hill. ISBN 9780071422949.

Pickering, K.D., Wines, K.R., Pariani, G.M., Franks, L.A., Yeh, J., Campbell, M.L., Finger, B.W., Verostko, C.E., Carrier, C., Gandhi, J.C. and Vega, L.M. (2001). Early results of an integrated water recovery system test. In: *31st International Conference On Environmental Systems*. SAE International, Orlando, FL.
Available at: http://dx.doi.org/10.4271/2001-01-2210

Powers, D. (2011). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–43. ISSN 22293981.

Reis, M. and Gins, G. (2017). Industrial Process Monitoring in the Big Data/Industry 4.0 Era: from Detection, to Diagnosis, to Prognosis. *Processes*, vol. 5, no. 3, p. 35. ISSN 22279717.

Rich, S.H. and Venkatasubramanian, V. (1987). Model-based reasoning in diagnostic expert systems for chemical process plants. *Computers & Chemical Engineering*, vol. 11, no. 2, pp. 111–122. ISSN 00981354.
Available at: http://dx.doi.org/10.1016/0098-1354(87)80012-1

Ricker, N.L. (1996). Decentralized control of the Tennessee Eastman challenge process. *Journal of Process Control*, vol. 6, no. 4, pp. 205–221. ISSN 09591524.
Available at: http://dx.doi.org/10.1016/0959-1524(96)00031-5

Roychoudhury, I., Biswas, G. and Koutsoukos, X. (2006). A Bayesian approach to efficient diagnosis of incipient faults. In: *17th International Workshop on the Principles of Diagnosis (DX-06)*, pp. 243–250. Peñaranda de Duero, Burgos.

Roychoudhury, I., Biswas, G. and Koutsoukos, X. (2008). Comprehensive diagnosis of continuous systems using dynamic Bayes nets. In: *19th International Workshop on the Principles of Diagnosis (DX-08)*. Blue Mountains, Sydney.

Roychoudhury, I., Biswas, G. and Koutsoukos, X. (2009). Distributed diagnosis of dynamic systems using dynamic Bayesian networks. In: *20th International Workshop on the Principles of Diagnosis (DX-09)*, pp. 329–336. Stockholm.

Roychoudhury, I., Biswas, G. and Koutsoukos, X. (2010). Distributed diagnosis in uncertain environments using dynamic Bayesian networks. In: *18th Mediterranean Conference on Control and Automation (MED'10)*, pp. 1531–1536. Institute of Electrical and Electronics Engineers (IEEE), Marrakech, Morocco.
Available at: http://dx.doi.org/10.1109/MED.2010.5547832

Sheridan, T.B. (1981). Understanding Human Error and Aiding Human Diagnostic Behaviour in Nuclear Power Plants. In: *Human Detection and Diagnosis of System Failures*, pp. 19–35. Springer, Boston, MA. ISBN 9781461592303.
Available at: http://dx.doi.org/10.1007/978-1-4615-9230-3_3

Thulasiraman, K. and Swamy, M.N.S. (1992). Directed Graphs. In: *Graphs: Theory and Algorithms*, chap. 5, pp. 97–125. John Wiley & Sons, Inc., Hoboken, NJ, USA. ISBN 9781118033104.
Available at: http://dx.doi.org/10.1002/9781118033104

Venkatasubramanian, V., Rengaswamy, R. and Kavuri, S.N. (2003*a*). A review of process fault detection and diagnosis — Part II: Qualitative models and search strategies. *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 313–326. ISSN 00981354.
Available at: http://dx.doi.org/10.1016/S0098-1354(02)00161-8

Venkatasubramanian, V., Rengaswamy, R., Kavuri, S.N. and Yin, K. (2003*b*). A review of process fault detection and diagnosis — Part I: Quantitative model-based methods. *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 293–311. ISSN 00981354.
Available at: http://dx.doi.org/10.1016/S0098-1354(02)00160-6

Venkatasubramanian, V., Rengaswamy, R., Kavuri, S.N. and Yin, K. (2003*c*). A review of process fault detection and diagnosis — Part III: Process history based methods. *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 327–346. ISSN 00981354.
Available at: http://dx.doi.org/10.1016/S0098-1354(02)00162-X

Verbert, K., Babuška, R. and De Schutter, B. (2017). Bayesian and Dempster-Shafer reasoning for knowledge-based fault diagnosis — A comparative study. *Engineering Applications of Artificial Intelligence*, vol. 60, pp. 136–150. ISSN 09521976.
Available at: http://dx.doi.org/10.1016/j.engappai.2017.01.011

Verron, S., Li, J. and Tiplica, T. (2010). Fault detection and isolation of faults in a multivariate process with Bayesian network. *Journal of Process Control*, vol. 20, no. 8, pp. 902–911. ISSN 09591524.
Available at: http://dx.doi.org/10.1016/j.jprocont.2010.06.001

Wakefield, B., Lindner, B., McCoy, J. and Auret, L. (2018). Monitoring of a simulated milling circuit: Fault diagnosis and economic impact. *Minerals Engineering*, vol. 120, pp. 132–151. ISSN 08926875.
Available at: http://dx.doi.org/10.1016/j.mineng.2018.02.007

Wasserman, L. (2004). *All of Statistics: A concise course in statistical inference*. Springer Texts in Statistics. Springer, New York, NY. ISBN 9781441923226.
Available at: http://dx.doi.org/10.1007/978-0-387-21736-9

Willsky, A.S. (1976). A survey of design methods for failure detection in dynamic systems. *Automatica*, vol. 12, no. 6, pp. 601–611. ISSN 00051098.
Available at: http://dx.doi.org/10.1016/0005-1098(76)90041-8

Yoshikawa, H. and Zhang, Z. (2014). *Progress of Nuclear Safety for Symbiosis and Sustainability*. Springer, Tokyo, Japan. ISBN 9784431546092.
Available at: http://dx.doi.org/10.1007/978-4-431-54610-8

Yu, J. and Rashid, M.M. (2013). A novel dynamic Bayesian network-based networked process monitoring approach for fault detection, propagation identification, and root cause diagnosis. *American Institute of Chemical Engineers (AIChE) Journal*, vol. 59, no. 7, pp. 2348–2365. ISSN 00011541.

Zhang, Z. and Dong, F. (2014). Fault detection and diagnosis for missing data systems with a three time-slice dynamic Bayesian network approach. *Chemometrics and Intelligent Laboratory Systems*, vol. 138, pp. 30–40. ISSN 01697439.
Available at: http://dx.doi.org/10.1016/j.chemolab.2014.07.009

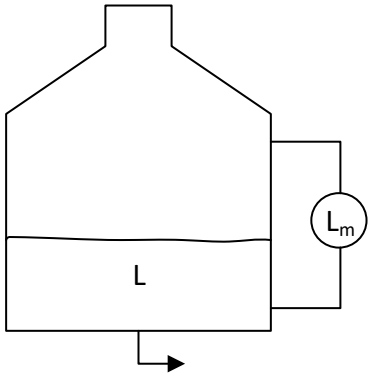
Zhao, Y., Xiao, F. and Wang, S. (2013). An intelligent chiller fault detection and diagnosis methodology using Bayesian belief network. *Energy and Buildings*, vol. 57, pp. 278–288. ISSN 03787788.
Available at: http://dx.doi.org/10.1016/j.enbuild.2012.11.007

Appendix A

Probabilistic Fault Diagnosis: Draining liquid level example

This demonstration considers a single component fault in a simple system and aims to showcase the inner workings of our DBN-based PD approach.

Consider the simple scenario in which a single tank is filled with water up to 50% of its maximum capacity and drained at a rate directly proportional to the current level in the tank. A level sensor measures current level of water in the tank and has a tendency to report noisy and/or inaccurate measurements of the actual level of water in the tank. This system was first introduced on page 46, but its schematic is replicated below for ease of reference.



Replicate of Figure 3.1: Schematic for draining liquid level example. L is the actual level of liquid in the tank and L_m is its measured value.

A.1 Start up and one time slice inference

Data collection

The data for the one tank system was generated synthetically based on the assumption that the liquid level in the tank decreases by $0.1L$ after each time slice. The sensor accurately measures the actual liquid level with a fair amount of noise up to time slice 3, after which it malfunctions and reads the last measured value for every subsequent time slice. Table A.1 shows the data used in this example.

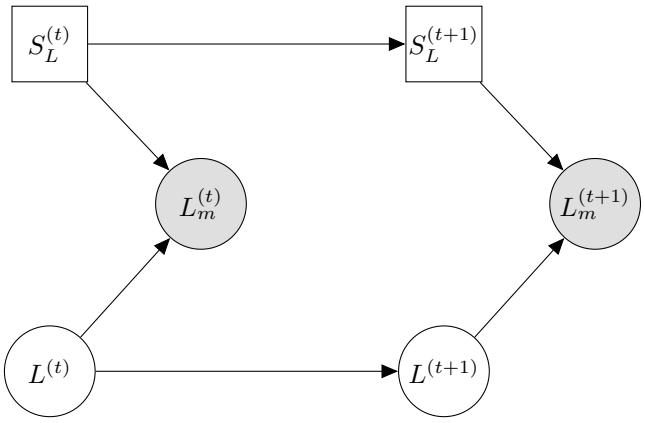
Table A.1: Process data for the draining liquid level example.

Time Slice	0	1	2	3	4	5	6	7	8
Actual	0.5	0.45	0.4050	0.3645	0.3281	0.2952	0.2657	0.2391	0.2152
Measured		0.46	0.4	0.4	0.4	0.4	0.4	0.4	0.4

It is also assumed that the sensor has a known variance of 0.001 when functioning *normally* and 0.03 when *faulty*.

DBN modelling

The structure of the DBN for this system was previously described and shown on page 47 and is replicated below for ease of reference.



Replicate of Figure 3.2: 2-TBN for draining liquid level example. L is the actual level of liquid in the tank and L_m is its measured value and S_L is the status of the sensor.

The DBN is initialised with the following CPDs:

Table A.2: CPDs for draining liquid level example DBN in Figure 3.2.

$S_L^{(0)}$	$\Pr\left(S_L^{(0)}\right)$
normal	0.9
faulty	0.1

$$\Pr\left(L^{(0)}\right) = \mathcal{N}\left(\mu_L = 0.5, \sigma_L^2 = 1\text{e-}4\right)$$

$$\Pr\left(L_m^{(t)} | L^{(t)}, S_L^{(t)} = \textit{normal}\right) = \mathcal{N}\left(\mu_{L_m} = L^{(t)}, \sigma_{L_m}^2 = 0.001 + \sigma_{L^{(t)}}^2\right)$$

$$\Pr\left(L_m^{(t)} | L^{(t)}, S_L^{(t)} = \textit{faulty}\right) = \mathcal{N}\left(\mu_{L_m} = L^{(t)}, \sigma_{L_m}^2 = 0.03 + \sigma_{L^{(t)}}^2\right)$$

$S_L^{(t)}$	$S_L^{(t+1)}$	$\Pr\left(S_L^{(t+1)} \mid S_L^{(t)}\right)$
normal	*normal*	0.8
faulty	*normal*	0.1
normal	*faulty*	0.2
faulty	*faulty*	0.9

$$\Pr\left(L^{(t+1)} | L^{(t)}\right) = \mathcal{N}\left(\mu_L = 0.9L^{(t)}, \sigma_L^2 = 1\text{e-}6 + 0.9 \times \sigma_{L^{(t)}}^2 \times 0.9\right)$$

Initial hypothesis

Approximate belief states are used to track process behaviour over time. The initial belief state typically consists of a single hypothesis describing part or all of the initial process state probability distributions. This includes a single multivariate Gaussian over all unobserved PVs and a discrete probability table over all process component statuses. The initial hypothesis can be obtained from the initial BN, $\mathcal{B}_0$, i.e. the JPD over all RVs in time slice 0. In this example the initial hypothesis contains a Gaussian over L and a discrete probability table over S_L. Recall that L_m is an observed RV and therefore is not part of the belief state or $\mathbf{Q}$.

One time slice inference

Since the initial hypothesis represents the belief state at time slice 0, the time slice 0 to 1 transitions can be computed from the initial hypothesis. This involves propagating the hypothesis through the 2-TBN to create a set of possible process behaviours depending on the transition of process component statuses from one time slice to the next. The transitions differently affect expected probability distributions for observed RVs at time slice 1 and therefore the transitions have different probabilities for being correct given their prior probability and observations of the process at time slice 1. Furthermore, the transitions also differently affect expected probability distributions for the unobserved RVs at time slice 1. Given observations of the process at time slice 1, these distributions describe different beliefs about the distributions of the unobserved RVs in the process at time slice 1. In order to determine which transitions should be combined with one another, the posterior probability distributions for the unobserved RVs at time slice 1 are compared for similarity according to Algorithm 1 on page 59. Upon finishing the inference procedure, each subset of transitions is

combined into a separate hypothesis and stored in the time slice 1 belief network.

Each of these steps are further detailed below in the context of the draining liquid level case study. Parameters for Algorithm 1 are set to $c = 0.005$ and $K = 2$ and no inference engine enhancements are applied. Also note that no discrete RVs are observed in this example and therefore $\mathbf{X}_\Delta$ and $\mathbf{Q}_\Delta$ can be used interchangeably, but for the sake of clarity and conciseness discrete RVs are referred to by $\mathbf{X}_\Delta$ here.

Computing $\Pr\left(\mathbf{X}_\Delta^{(t)}, \mathbf{X}_\Delta^{(t+1)}\right)$:

S_L is the only discrete RV in this case study and is also not observed. Since S_L has two possible assignments, there will be a total of four transitions to compute from one time slice to the next. First the JPD over the discrete RVs in time slice t and $t+1$ is computed by way of

$$\Pr\left(S_L^{(t)}, S_L^{(t+1)}\right) = \Pr\left(S_L^{(t+1)} | S_L^{(t)}\right) \times \Pr\left(S_L^{(t)}\right). \tag{A.1}$$

From $t = 0$ to $t = 1$ this produces the probability distribution:

$S_L^{(0)}$	$S_L^{(1)}$	$\Pr\left(S_L^{(0)}, S_L^{(1)}\right)$
normal	*normal*	$0.8 \times 0.9 = 0.72$
faulty	*normal*	$0.1 \times 0.1 = 0.01$
normal	*faulty*	$0.2 \times 0.9 = 0.18$
faulty	*faulty*	$0.9 \times 0.1 = 0.09$

Computing $\Pr\left(\mathbf{O}_\Gamma^{(t+1)} = \mathbf{o}_\Gamma^{(t+1)} | \mathbf{X}_\Delta^{(t)}, \mathbf{X}_\Delta^{(t+1)}\right)$:

Next the expected probability distribution for the observed RVs, i.e. L_m, in time slice 1 needs to be determined. Since the RVs in time slice t would be marginalised anyway, calculations show direct computation of the Gaussians over the RVs in time slice $t+1$ rather than the JPD and its marginal. This will not affect the results, but makes the explanation more concise and easier to follow.

Recall that $L^{(t+1)}$ is only dependent on $L^{(t)}$ and $L_m^{(t+1)}$ is only dependent on $\{S_L^{(t+1)}, L^{(t+1)}\}$. Therefore, it is better to first compute $\Pr\left(L^{(1)}\right)$ and use it to later determine $\Pr\left(L^{(1)}, L_m^{(1)} | S_L^{(1)}\right)$. This first part is done by propagating $\Pr\left(L^{(0)}\right)$ through the 2-TBN and marginalising it out of the resulting JPD over $L^{(0)}$ and $L^{(1)}$, i.e. according to Equation 2.40:

$$\Pr\left(L^{(1)}\right): \quad \begin{array}{l|l} \mu_L = \beta_0 + \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{\mu} & \sigma_L^2 = \sigma_0^2 + \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{\Sigma}\boldsymbol{\beta} \\ = 0 + (0.9)(0.5) & = \text{1e-6} + (0.9)(\text{1e-4})(0.9) \\ = 0.45 & = 0.000082 \end{array}$$

The same Equation 2.40 can be used to determine $\Pr\left(L_m^{(1)}|S_L^{(1)}\right)$, the expected probability distribution for the observed RVs given various process operating modes. In this case the JPD over $L^{(1)}$ and $L_m^{(1)}$ is computed from $\Pr\left(L^{(1)}\right)$ and the BN (i.e. CPD of $L_m^{(1)}$), then $L^{(1)}$ is marginalised out from that JPD. Note that there are two JPDs because of the dependence of $L_m^{(1)}$ on $S_L^{(1)}$, i.e. a JPD needs to be computed for each assignment to $S_L^{(1)}$. Once again using Equation 2.40, for each assignment to $S_L^{(1)}$, $\Pr\left(L_m^{(1)}|S_L^{(1)}\right)$ is computed as:

$$\Pr\left(L_m^{(1)}|S_L^{(1)}=\textit{normal}\right): \quad \begin{aligned}\mu_{L_m} &= 0+(1)(0.45)\\ &= 0.45\end{aligned} \;\Big|\; \begin{aligned}\sigma^2_{L_m} &= 0.001+(1)(0.000082)(1)\\ &= 0.001082\end{aligned}$$

$$\Pr\left(L_m^{(1)}|S_L^{(1)}=\textit{faulty}\right): \quad \begin{aligned}\mu_{L_m} &= 0+(1)(0.45)\\ &= 0.45\end{aligned} \;\Big|\; \begin{aligned}\sigma^2_{L_m} &= 0.03+(1)(0.000082)(1)\\ &= 0.030082\end{aligned}$$

Each of these Gaussians are associated with the transitions which agree on their assignment to $S_L^{(1)}$, i.e. $\Pr\left(L_m^{(1)}|S_L^{(1)}=n\right)$ is associated with $\Pr\left(S_L^{(0)}=n, S_L^{(1)}=n\right)$ and $\Pr\left(S_L^{(0)}=f, S_L^{(1)}=n\right)$.

The likelihood of an observation for a given Gaussian can be computed by substituting the observation for $L_m^{(1)}$ into the Gaussian function, Equation 2.35. Doing this results in:

$$\begin{aligned}\Pr\left(L_m^{(1)}=0.46|S_L^{(1)}=\textit{normal}\right) &= \frac{1}{\sqrt{2\pi(0.001082)}}\exp\left(-\frac{(0.46-0.45)^2}{2(0.001082)}\right)\\ &= 11.5805\end{aligned} \tag{A.2}$$

and,

$$\begin{aligned}\Pr\left(L_m^{(1)}=0.46|S_L^{(1)}=\textit{faulty}\right) &= \frac{1}{\sqrt{2\pi(0.030082)}}\exp\left(-\frac{(0.46-0.45)^2}{2(0.030082)}\right)\\ &= 2.2963.\end{aligned} \tag{A.3}$$

Computing $\Pr\left(\mathbf{X}_\Delta^{(t)}, \mathbf{X}_\Delta^{(t+1)}|\mathbf{O}_\Gamma^{(t+1)}=\mathbf{o}_\Gamma^{(t+1)}\right)$:
These likelihoods can be used to determine the posterior probabilities of the transitions using Equation 2.31. In this case the likelihood function is $\Pr\left(L_m^{(1)}=0.46|S_L^{(1)}\right)$, the prior is $\Pr\left(S_L^{(0)}, S_L^{(1)}\right)$, and the normalisation constant is the sum of their products

for each set of assignments to $\{S_L^{(0)}, S_L^{(1)}\}$. The resulting *a posteriori* probability distribution over the set of transitions is then:

$S_L^{(0)}$	$S_L^{(1)}$	$\Pr\left(L_m^{(1)} = 0.46 \mid S_L^{(1)}\right)$	$\Pr\left(S_L^{(0)}, S_L^{(1)}\right)$	$\Pr\left(S_L^{(0)}, S_L^{(1)} \mid L_m^{(1)} = 0.46\right)$
normal	*normal*	11.5805	0.72	0.9189
faulty	*normal*	11.5805	0.01	0.0128
normal	*faulty*	2.2963	0.18	0.0456
faulty	*faulty*	2.2963	0.09	0.0228

Computing $\Pr\left(\mathbf{Q}_\Gamma^{(t+1)} \mid \mathbf{X}_\Delta^{(t)}, \mathbf{X}_\Delta^{(t+1)}, \mathbf{O}_\Gamma^{(t+1)} = \mathbf{o}_\Gamma^{(t+1)}\right)$:

The multivariate Gaussians over $\{L^{(1)}, L_m^{(1)}\}$ associated with each of these transitions can be determined using Equation 2.42, $\Pr\left(L^{(1)}\right)$, and $\Pr\left(L_m^{(1)} \mid S_L^{(1)}\right)$. Equation 2.42 simply determines the covariance between $L^{(1)}$ and $L_m^{(1)}$ given $S_L^{(1)}$ and is used to complete the covariance matrix for each multivariate Gaussian, resulting in:

$$\Pr\left(L^{(1)}, L_m^{(1)} \mid S_L^{(1)} = \textit{normal}\right): \quad \mu_{L,L_m} = \begin{bmatrix} 0.45 \\ 0.45 \end{bmatrix} \;\middle|\; \sigma^2_{L,L_m} = \begin{bmatrix} 0.000082 & 0.000082 \\ 0.000082 & 0.001082 \end{bmatrix}$$

$$\Pr\left(L^{(1)}, L_m^{(1)} \mid S_L^{(1)} = \textit{faulty}\right): \quad \mu_{L,L_m} = \begin{bmatrix} 0.45 \\ 0.45 \end{bmatrix} \;\middle|\; \sigma^2_{L,L_m} = \begin{bmatrix} 0.000082 & 0.000082 \\ 0.000082 & 0.030082 \end{bmatrix}$$

$\Pr\left(L^{(1)} \mid L_m^{(1)} = 0.46, S_L^{(1)}\right)$ is required to determine which transitions should be combined with one another to form hypotheses for the time slice 1 belief state. Given the above multivariate Gaussians, it easy to compute this probability distribution by conditioning on $L_m^{(1)} = 0.46$ for each assignment to $S_L^{(1)}$. This is done according to Equations 2.44 and 2.45 where $L^{(1)}$ represents $\mathbf{Q}$ and $L_m^{(1)}$ represents $\mathbf{O}$, resulting in:

$$\begin{aligned} \Pr\left(L^{(1)} \mid L_m^{(1)} = 0.46, S_L^{(1)} = \textit{normal}\right) &: \mathcal{N}\left(\mu_L = 0.4508, \sigma_L^2 = 0.00007586\right) \\ \Pr\left(L^{(1)} \mid L_m^{(1)} = 0.46, S_L^{(1)} = \textit{faulty}\right) &: \mathcal{N}\left(\mu_L = 0.45, \sigma_L^2 = 0.000081776\right) \end{aligned} \tag{A.4}$$

Note that there are four transitions, but only two unique Gaussians because of the independence structure of the BN. Once again each Gaussian is associated with the transitions which agree on their assignment to $S_L^{(1)}$.

Comparing similarities between transitions:

Since there are only two unique Gaussians, only one similarity value needs to be computed, viz. the similarity between:

$$\mathcal{N}_1 : \Pr\left(L^{(1)} \mid L_m^{(1)} = 0.46, S_L^{(1)} = \textit{normal}\right)$$

and

$$\mathcal{N}_2 : \Pr\left(L^{(1)}|L_m^{(1)} = 0.46, S_L^{(1)} = \textit{faulty}\right).$$

Note that we do not interpret the Kullback-Leibler divergences between $\mathcal{N}_1$ and $\mathcal{N}_2$ directly as these values cannot be used as distance measures. Thus we only interpret $J(\mathcal{N}_1, \mathcal{N}_2)$ – by comparison with c as presented later.

Using Equation 3.10, $D_{KL}(\mathcal{N}_1||\mathcal{N}_2)$ may be computed as:

$$\begin{aligned} D_{KL}(\mathcal{N}_1||\mathcal{N}_2) &= \frac{1}{2}\left(\frac{0.00007586}{0.000081776} - 1\right) + \frac{1}{2}\left(\log\frac{0.000081776}{0.00007586} + \frac{(0.45 - 0.4508)^2}{0.000081776}\right) \\ &= 0.0053. \end{aligned} \tag{A.5}$$

Similarly, $D_{KL}(\mathcal{N}_2||\mathcal{N}_1)$ may be computed as:

$$\begin{aligned} D_{KL}(\mathcal{N}_2||\mathcal{N}_1) &= \frac{1}{2}\left(\frac{0.000081776}{0.00007586} - 1\right) + \frac{1}{2}\left(\log\frac{0.00007586}{0.000081776} + \frac{(0.4508 - 0.45)^2}{0.00007586}\right) \\ &= 0.0057. \end{aligned} \tag{A.6}$$

$J(\mathcal{N}_1, \mathcal{N}_2)$ can then be calculated using Equation 3.11:

$$\begin{aligned} J(\mathcal{N}_1, \mathcal{N}_2) &= \frac{1}{2}(0.0053 + 0.0057) \\ &= 0.0055. \end{aligned} \tag{A.7}$$

Combining transitions and creating hypotheses:
Since $J(\mathcal{N}_1, \mathcal{N}_2)$ is greater than c, the two Gaussians are deemed dissimilar. Therefore, only the transitions with the same Gaussians, due to the independence structure in the DBN, are combined with one another. In accordance with Algorithm 1, this creates two hypotheses (the maximum) which populate the belief network at time slice 1. The CPDs of the belief network are thus:

Table A.3: Belief network CPDs at time slice 1, given the observation $L_m^{(1)} = 0.46$, for draining liquid level example.

$\mathbf{q}_\Delta$	S_L	$\Pr\left(\Delta_{\text{bel}}^{(1)}, H_{\text{bel}}^{(1)} = 1\right)$	$\Pr\left(\Delta_{\text{bel}}^{(1)}, H_{\text{bel}}^{(1)} = 2\right)$	$\Pr\left(\Delta_{\text{bel}}^{(1)}\right)$
1	*normal*	0.9317	0	0.9317
2	*faulty*	0	0.0684	0.0684
SUM		0.9317	0.0684	1

$$\Pr\left(\Gamma_{\text{bel}}^{(1)}|H_{\text{bel}}^{(1)} = 1\right) : \mathcal{N}(0.4508, 0.00007586)$$
$$\Pr\left(\Gamma_{\text{bel}}^{(1)}|H_{\text{bel}}^{(1)} = 2\right) : \mathcal{N}(0.45, 0.000081776)$$

Computing $\tilde{\Pr}\left(\mathbf{Q}_{\Delta}^{(t+1)}, \mathbf{Q}_{\Gamma}^{(t+1)} | \mathbf{O}^{(t+1)} = \mathbf{o}^{(t+1)}\right)$:
The approximate belief state at time slice 1 can be computed by marginalising out $H_{\text{bel}}^{(1)}$ from the time slice 1 belief network. The distribution over $\Delta_{\text{bel}}^{(1)}$ had already been computed in Table A.3 as the sum of entries in $\Pr\left(\Delta_{\text{bel}}^{(1)}, H_{\text{bel}}^{(1)}\right)$ which agree on their assignment to $\Delta_{\text{bel}}^{(1)}$. This leaves the distribution over $\Gamma_{\text{bel}}^{(1)}$ to be computed. This is done by considering $\Pr\left(\Gamma_{\text{bel}}^{(1)} | H_{\text{bel}}^{(1)}\right)$ as a hybrid mixture of Gaussians which can be collapsed using $\Pr\left(H_{\text{bel}}^{(1)}\right)$ as the set of weights. Doing so produces the Gaussian:

$$\Pr\left(\Gamma_{\text{bel}}^{(1)}\right): \quad \begin{aligned} \tilde{\boldsymbol{\mu}} &= \sum_{i=1}^{n} w_i \boldsymbol{\mu}_i \\ &= (0.9317)\,(0.4508) \\ &\quad + (0.0684)\,(0.45) \\ &= 0.4508 \end{aligned} \qquad \begin{aligned} \tilde{\boldsymbol{\Sigma}} &= \sum_{i=1}^{n} w_i \boldsymbol{\Sigma}_i \\ &\quad + \sum_{i=1}^{n} w_i \left(\boldsymbol{\mu}_i - \tilde{\boldsymbol{\mu}}\right)\left(\boldsymbol{\mu}_i - \tilde{\boldsymbol{\mu}}\right)^{\mathrm{T}} \\ &= (0.9317)\,(0.00007586) \\ &\quad + (0.0684)\,(0.000081776) \\ &\quad + (0.9317)\,(0.4508 - \tilde{\boldsymbol{\mu}})^2 \\ &\quad + (0.0684)\,(0.45 - \tilde{\boldsymbol{\mu}})^2 \\ &= 0.000076315 \end{aligned}$$

This Gaussian is associated with all assignments to $\Delta_{\text{bel}}^{(1)}$. Therefore this Gaussian $\Pr\left(\Gamma_{\text{bel}}^{(1)}\right)$ completes the approximate belief state for time slice 1.

A.2 Process state tracking performance

Tracking the process state is accomplished by using the approximate belief state for time slice 1 and following the one time slice inference procedure, then doing the same using the the approximate belief state for time slice 2 and so forth. Since the belief state contains only one PV and one process component's status in this case study, tracking of the inferred PV value versus actual process behaviour and component abnormality status can easily be represented on two time-series plots.

Figure A.1 shows how the inference engine tracked the actual liquid level in the tank over time. Notice that even though the sensor malfunctioned after time step 3, the inference engine was still able to reliably track the actual liquid level in the tank over time.

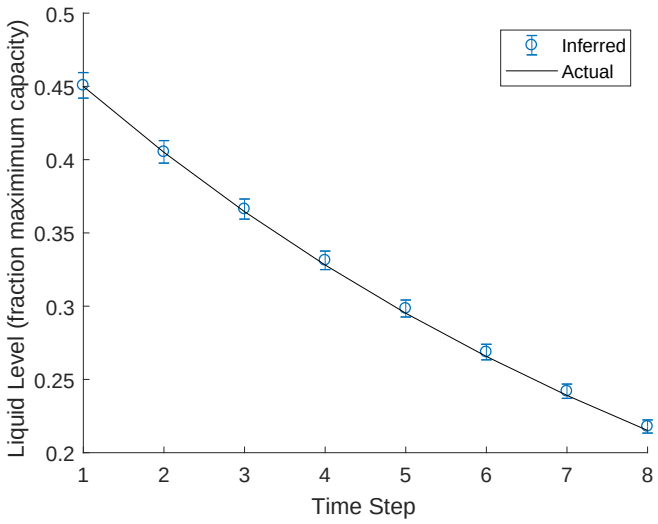


Figure A.1: Comparison of actual versus inferred liquid level over time.

This is because the PD approach detected the malfunction of the sensor after time slice 3, as shown in Figure A.2, which caused it to effectively discard the readings of the malfunctioned sensor and rely more on the predictions of its internal DBN model.

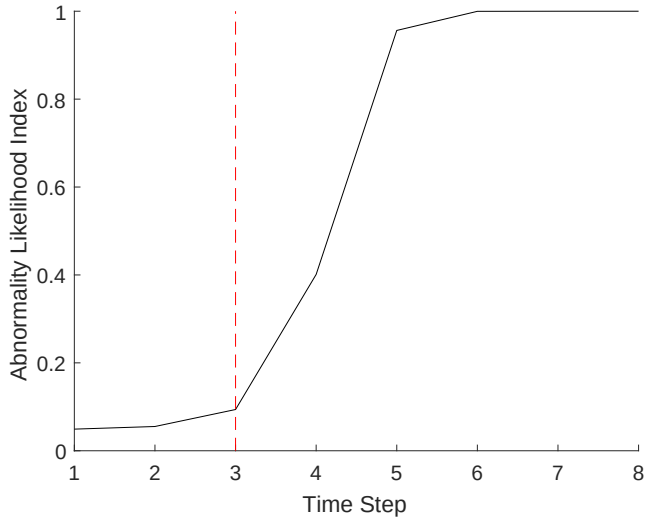


Figure A.2: Monitoring of process abnormality likelihood index over time. The dashed line indicates the start of fault conditions.

A.3 Non-linear liquid level drainage

In practice, natural liquid level drainage is not a linear function with respect to time. A more accurate depiction of the liquid level drainage in the single tank system may be

the non-linear function:

$$L^{(t+1)} = L^{(t)} - 0.1\sqrt{L^{(t)}} \tag{A.8}$$

The PD approach is equipped to handle non-linear CPDs by way of just-in-time linearisation as discussed in Section 3.4.2. In order to evaluate how well this approach works in practice, the DBN model for the liquid level drainage is changed to the non-linear function in Equation A.8.

The ability of the PD approach to track the actual liquid level was tested under two scenarios, one where it would observe an accurate reading of the liquid level in each time step and the other where it would observe a reading of 0 for the liquid level in each time step. These scenarios were labelled as the best and worst cases respectively. In these tests, the PD approach was limited to maintain only one hypothesis in its belief network and not combine any transitions with one another.

The results of the tests (Figure A.3) show that the PD approach was still able to accurately track the liquid level drainage with the non-linear function under both best and worst case scenarios. Furthermore, the best case scenario results maintained an average 0% error (in terms of deviation) from the actual liquid level with respect to time and the worst case scenario slowly increased in percentage error from the actual liquid level with respect to time as expected, due to lack of accurate, up-to-date, sensor readings.

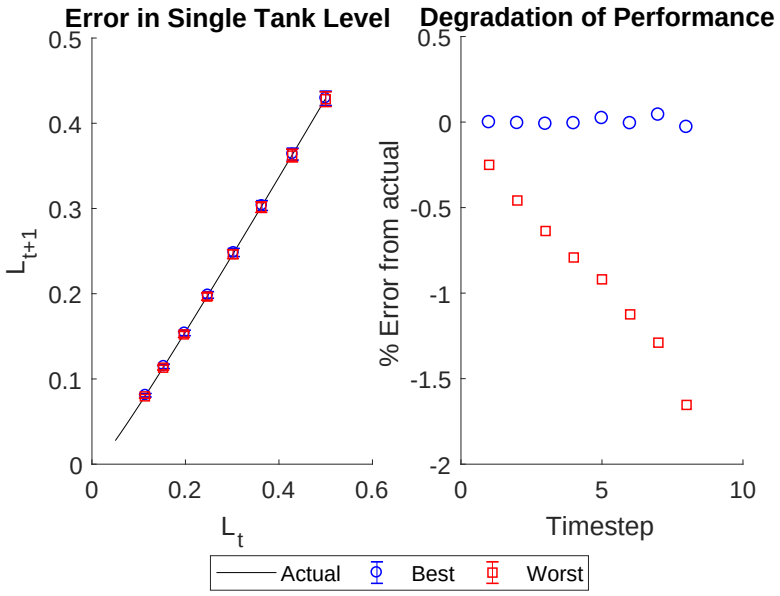


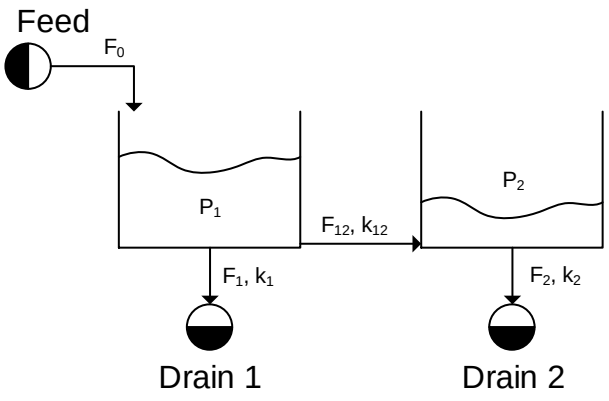
Figure A.3: Accuracy of estimating a non-linear liquid level drainage using the PD approach.

Appendix B

Two-tank System: Process modelling, Simulation, and DBN

B.1 Fundamental process model

Consider the two tank system shown in Figure 5.6.



Replicate of Figure 5.6. Two tanks in series.

Water flow F_0 is directed into tank 1 which is connected to tank 2 by a horizontal connecting pipe at their bottoms. The flows F_1 and F_2 drain the two tanks, respectively. The flow in the connecting pipe F_{12} is determined by the driving forces (pressure) P_1 and P_2 at the outlet of each tank respectively. The overall mass balances for each of the tanks are given by:

$$\rho\frac{\mathrm{d}V_1}{\mathrm{d}t} = \rho\left(F_0 - F_1 - F_{12}\right) \quad \text{and} \quad \rho\frac{\mathrm{d}V_2}{\mathrm{d}t} = \rho\left(F_{12} - F_2\right) \tag{B.1}$$

where V_1 and V_2 are the volumes of water in each tank and ρ is the density of water.

The volumes V_i are a product of the cross-sectional area, A_i, and height of water, h; in each tank i, namely:

$$V_i = A_i \times h_i \tag{B.2}$$

Each h_i, in turn, may be written as a function of hydrostatic pressure in a tank, namely:

$$h_i = \frac{P_i}{\rho g} \tag{B.3}$$

where ρ is the density of water and g is the gravitational acceleration constant.

Using Equation B.2, Equation B.2, and Equation B.3 yields the governing equations (see also Equation 5.4) for the model.

$$\begin{aligned} \frac{\mathrm{d}P_1}{\mathrm{d}t} &= \frac{\rho g}{A_1}\left(F_0 - F_1 - F_{12}\right) \\ \frac{\mathrm{d}P_2}{\mathrm{d}t} &= \frac{\rho g}{A_2}\left(F_{12} - F_2\right) \end{aligned} \tag{B.4}$$

The flow F_0 is given by some input forcing function which is initially assumed to be constant. The flows F_1, F_2 and F_{12} are modelled as flow through a constriction (Equation 2.15) and are the constitutive equations given by:

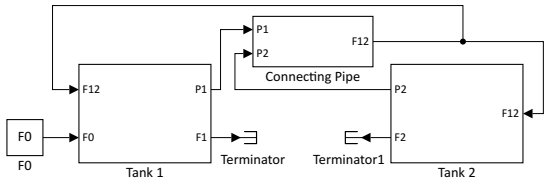
$$F_1 = k_1\sqrt{P_1} \quad \text{and} \quad F_2 = k_2\sqrt{P_2} \quad \text{and} \quad F_{12} = k_{12}\sqrt{|P_1 - P_2|} \times \operatorname{sign}(P_1 - P_2) \tag{B.5}$$

where k_1, k_2 and k_{12} are the constriction resistances of each pipe.

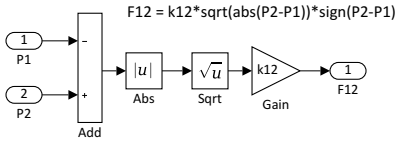
The fundamental process model for the two tank system consists of the governing Equations 5.4 and the constitutive Equations 5.5.

B.2 Simulation

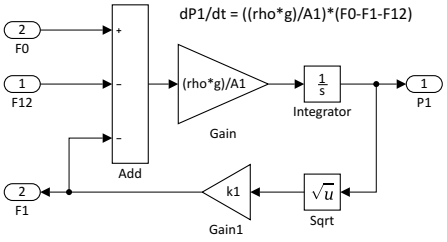
Figure B.1 shows the implementation of the two tank system process model in Simulink™ . The values of the constants are given in Table 5.1.



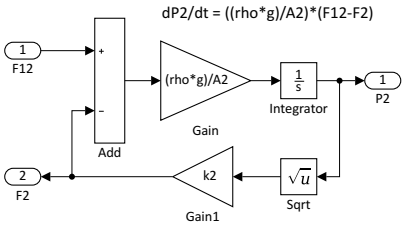
(a) Main view



(b) Connecting pipe



(c) Tank 1



(d) Tank 2

Figure B.1: Two tank system process model implementation in Simulink™ .

B.3 Dynamic Bayesian network

The ensuing subsections detail the development of the DBN for the two tank system.

B.3.1 Skeleton 2-TBN

Following Algorithm 3, the variables in Equations 5.4 and 5.5 are: P_1, P_2, F_1, F_2, and F_{12}. For Equations 5.5 an edge is made both in time slice t and $t+1$: (1) from P_1 to F_1, (2) from P_2 to F_2, and (3) from P_1 and P_2 to F_{12}. For Equation 5.4 an edge is made from across the time slices: (4) from F_1, F_{12}, and P_1 to P_1, and (5) from F_{12}, F_2 and P_2 to P_2. The result is the diagram in Figure B.2 where each of the edges are numbered in accordance with operations in the text above. Note that F_0 is subsumed by P_1 in this diagram in the same manner as in Lerner *et al.* (2000) for consistency between the two approaches.

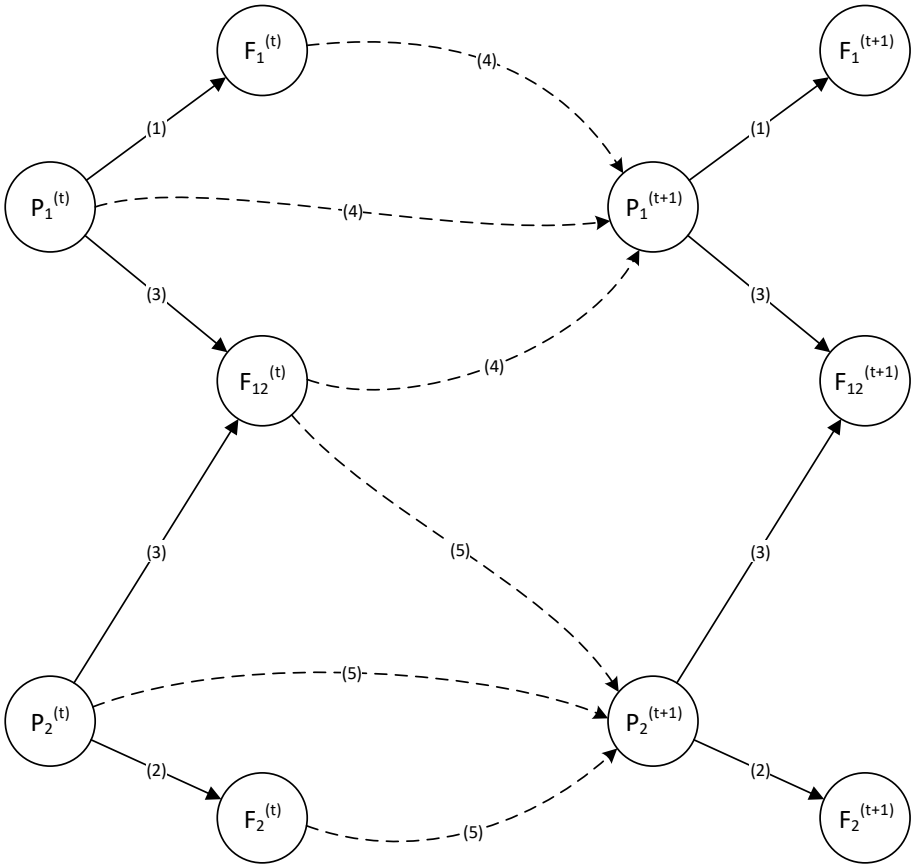


Figure B.2: 2-TBN for two tank system. Solid lines represent intra-time slice edges and dotted lines represent inter-time slice edges.

B.3.2 Adding sensor and fault models

Next, sensor models are added for P_1 and P_2 including fault modes for a normal and biased sensor. Additionally, a fault model is added for blockage of the connecting pipe between the two tanks by changing the model parameter k_{12}. This is shown Figure B.3.

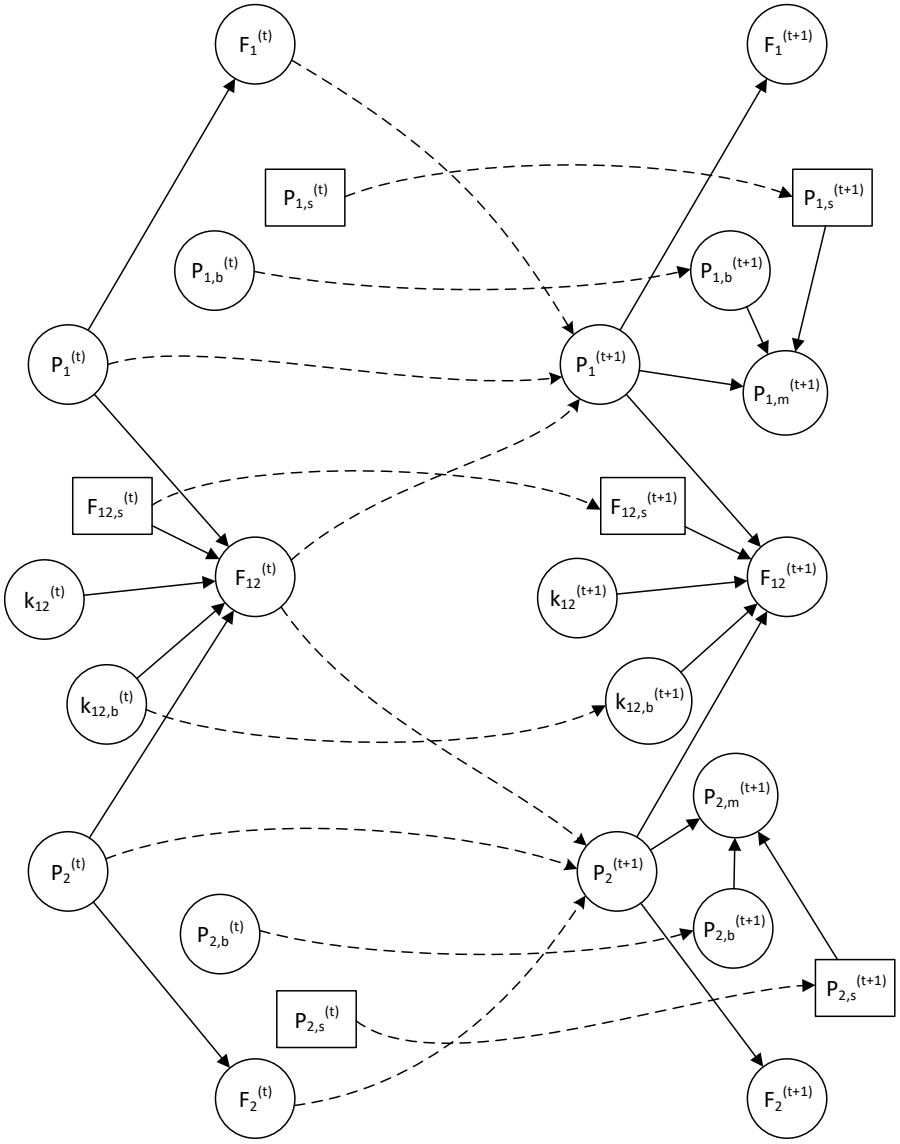


Figure B.3: DBN model for the two-tank system including sensor and fault models. The subscripts s, b, and m refer to status, bias, and measured RVs respectively. Note that this is not the complete structure of the 2-TBN used in this thesis which includes sensor models for all flows (including F_0) as well. This was done so as to not make the figure too busy.

Appendix C

Two-tank System: Simulated data

C.1 Configuration sN

C.1.1 aConn fault

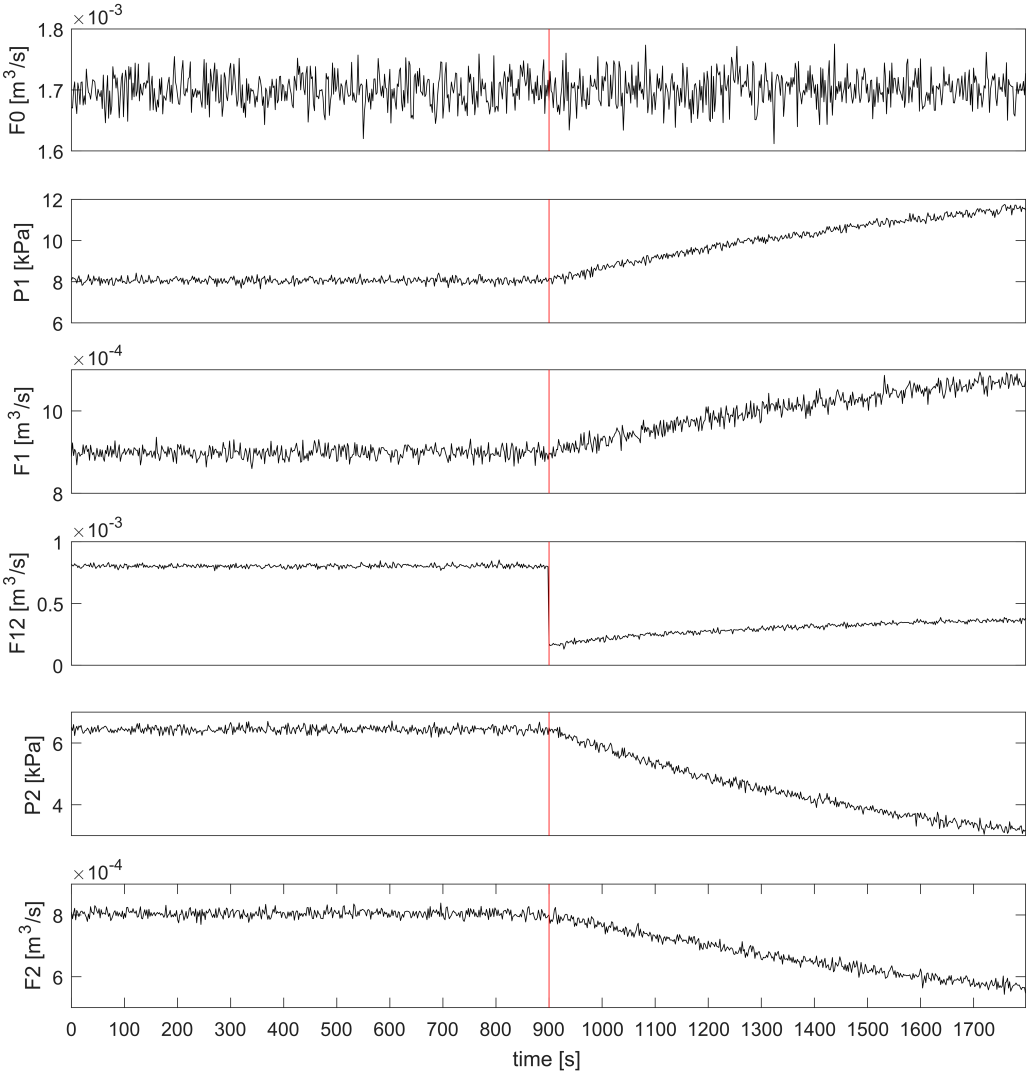


Figure C.1: Two-tank system process data: Configuration sN with aConn fault manifesting after 900 seconds.

C.1.2 iConn fault

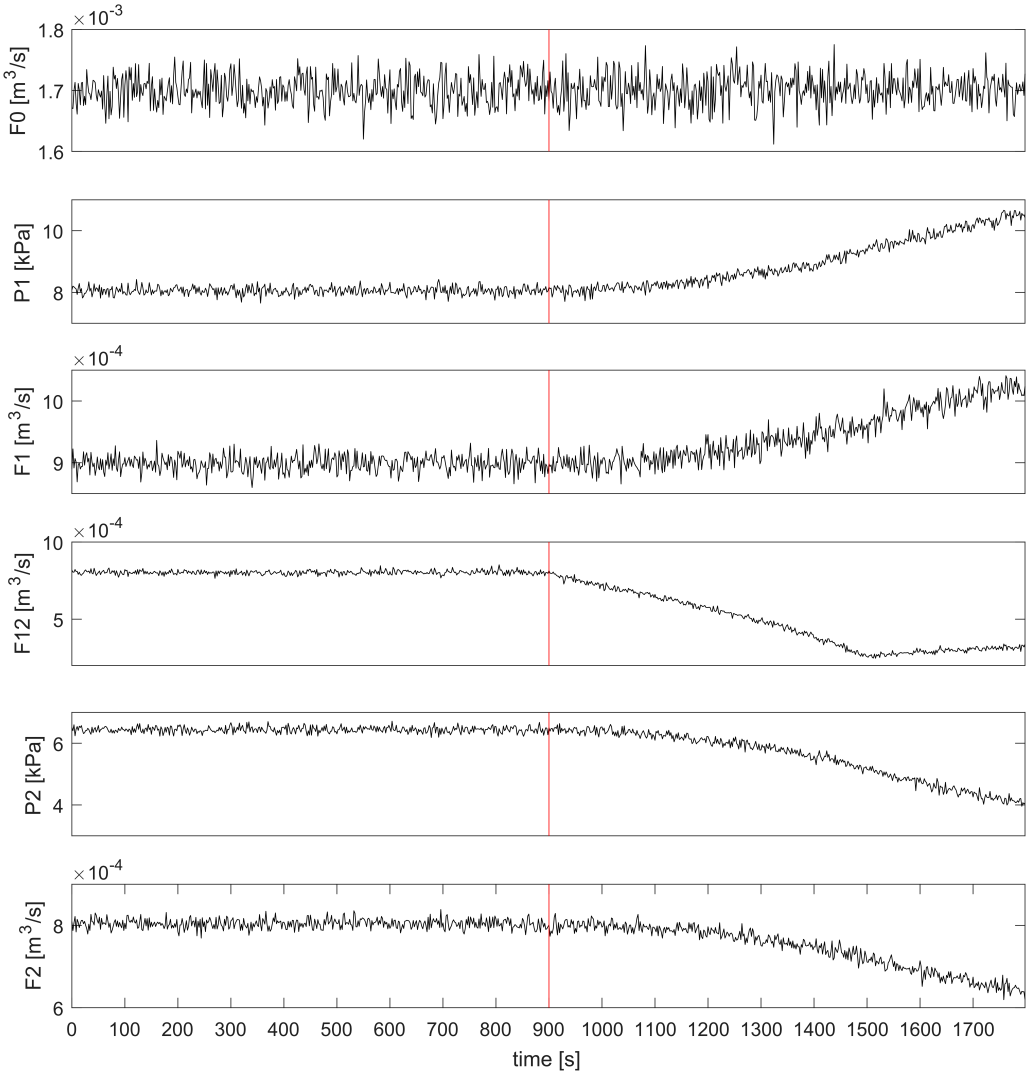


Figure C.2: Two-tank system process data: Configuration sN with iConn fault manifesting after 900 seconds.

C.1.3 P1s fault

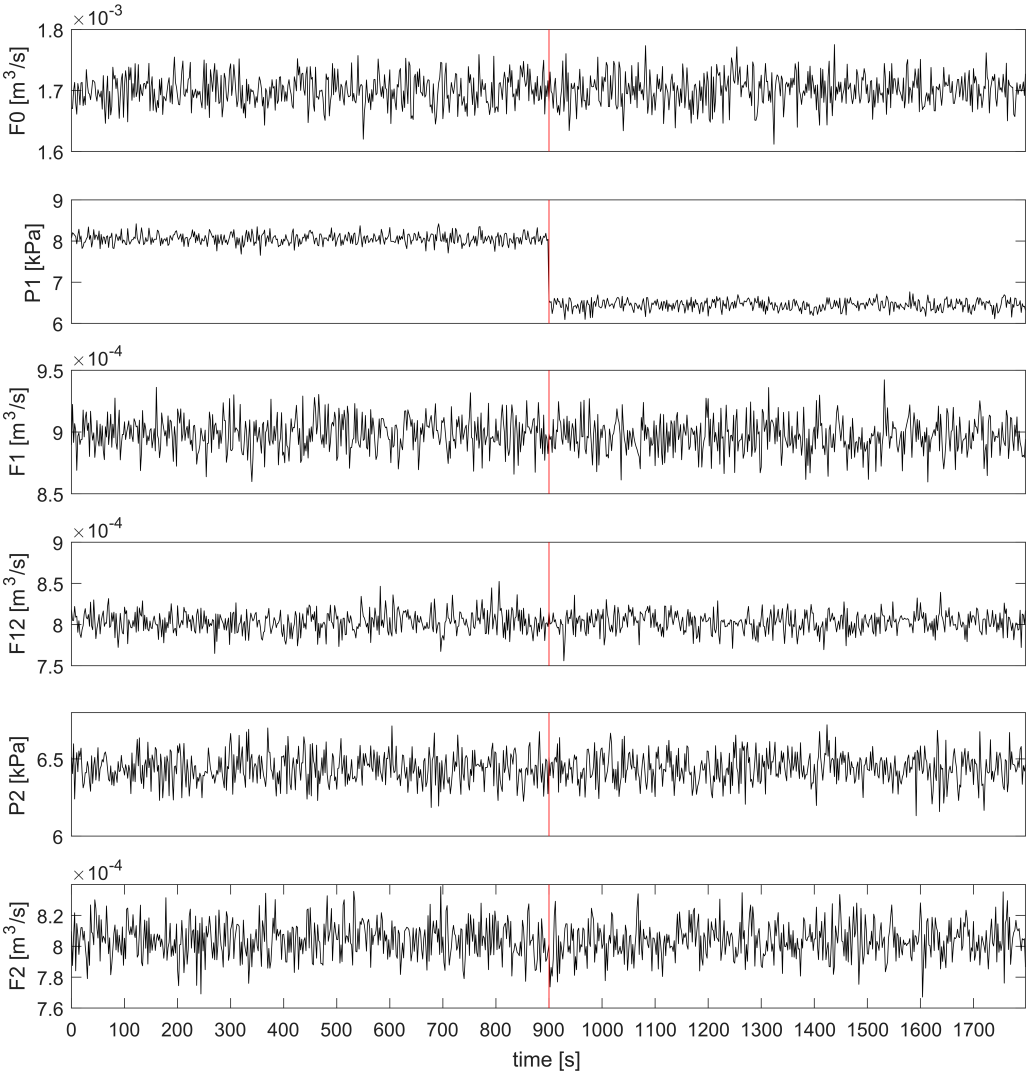


Figure C.3: Two-tank system process data: Configuration sN with P1s fault manifesting after 900 seconds.

C.1.4 P2s fault

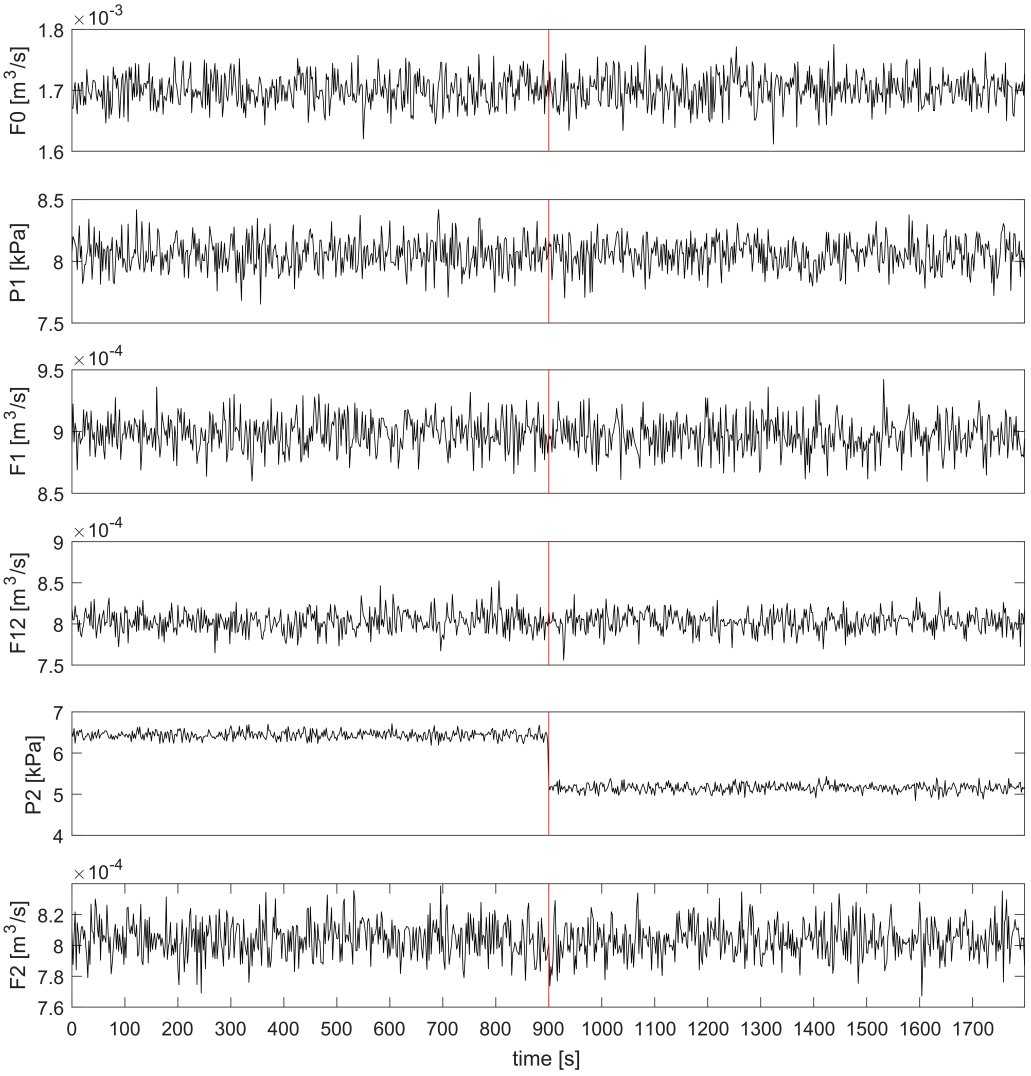


Figure C.4: Two-tank system process data: Configuration sN with P2s fault manifesting after 900 seconds.

C.2 Configuration N

C.2.1 aConn fault

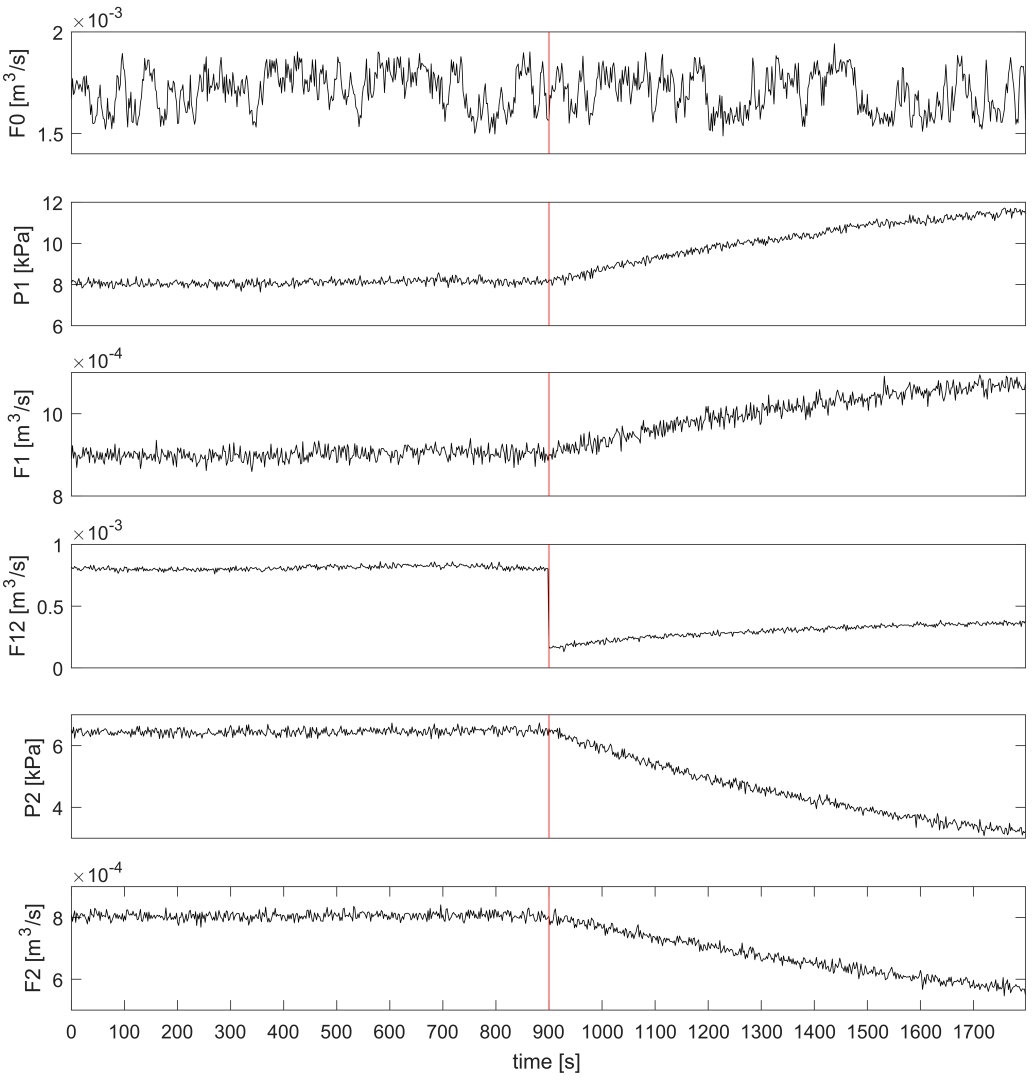


Figure C.5: Two-tank system process data: Configuration N with aConn fault manifesting after 900 seconds.

C.2.2 iConn fault

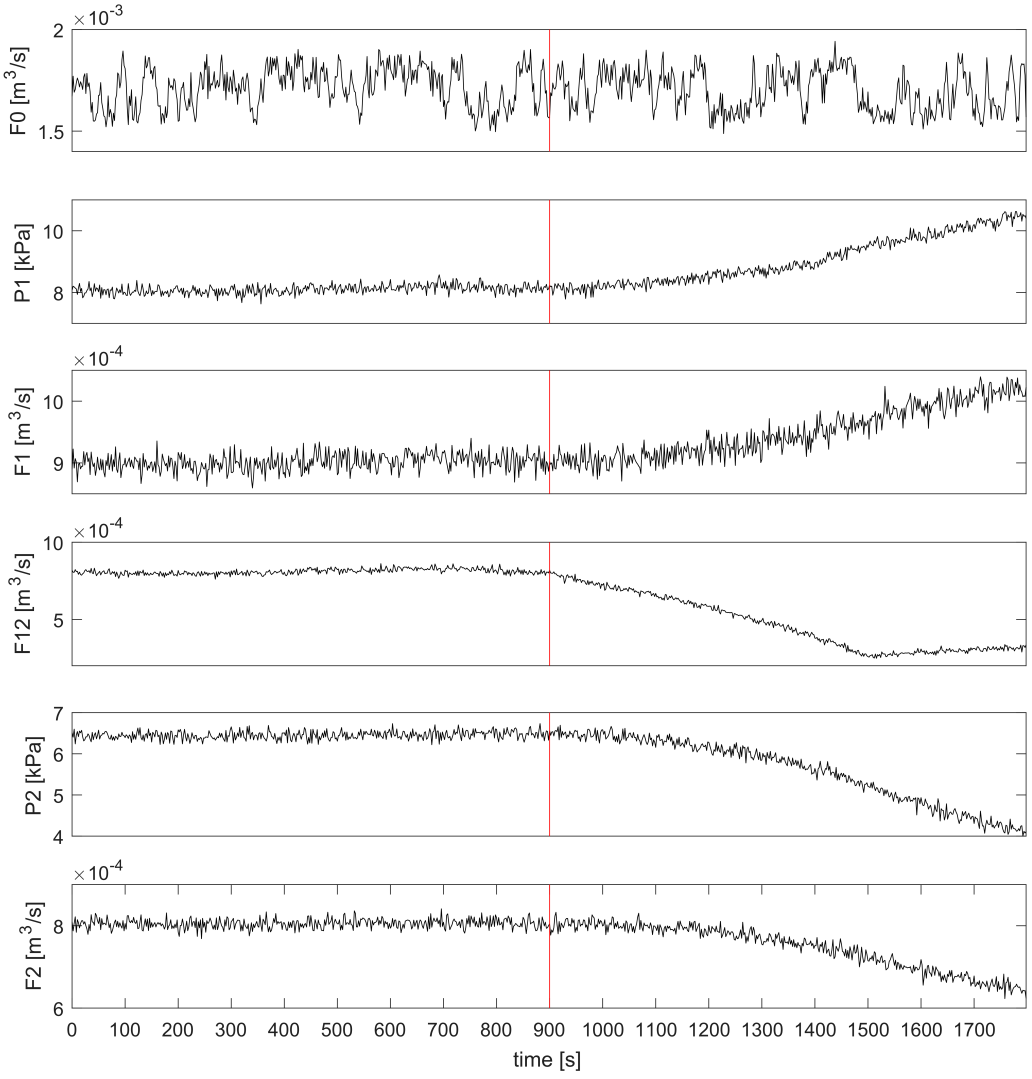


Figure C.6: Two-tank system process data: Configuration N with iConn fault manifesting after 900 seconds.

C.2.3 P1s fault

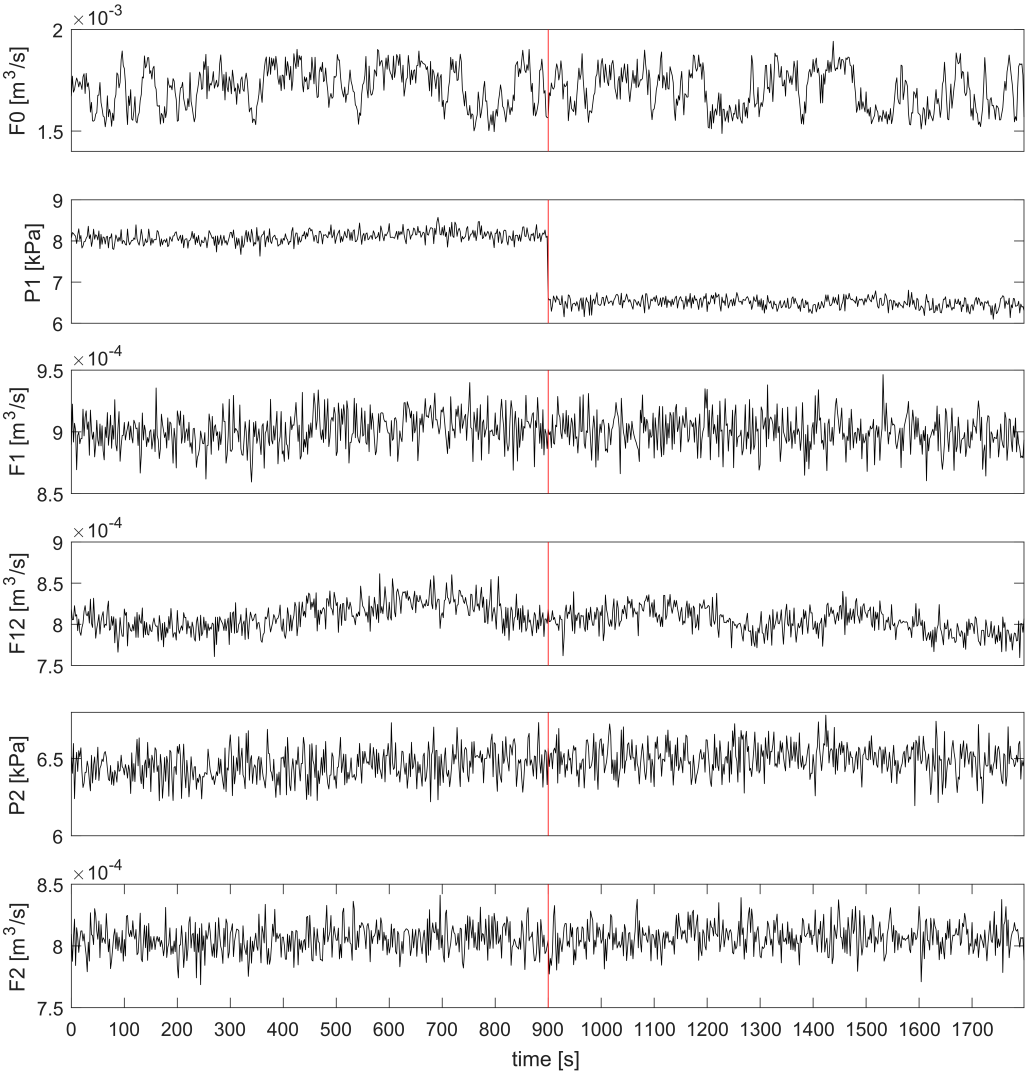


Figure C.7: Two-tank system process data: Configuration N with P1s fault manifesting after 900 seconds.

C.2.4 P2s fault

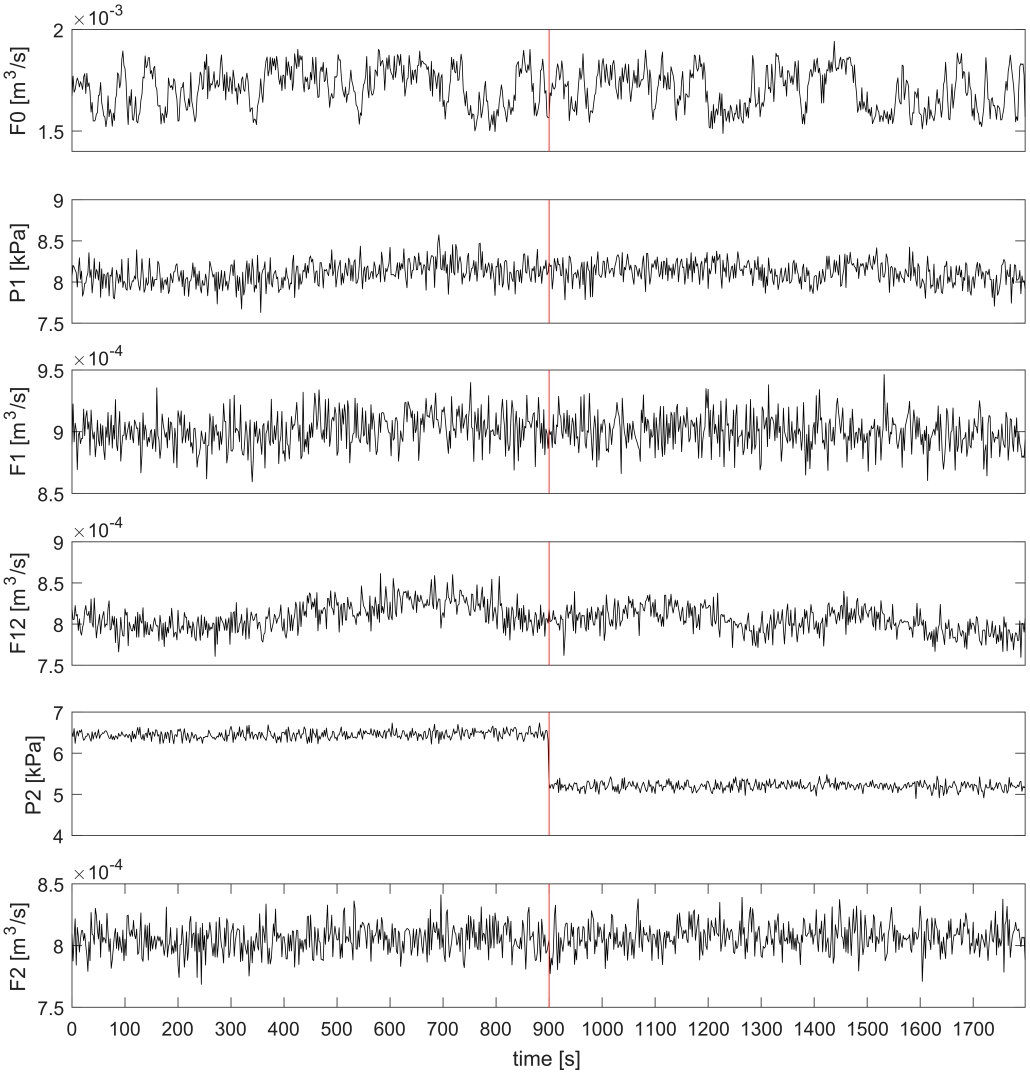


Figure C.8: Two-tank system process data: Configuration N with P2s fault manifesting after 900 seconds.

C.3 Configuration NC

C.3.1 aConn fault

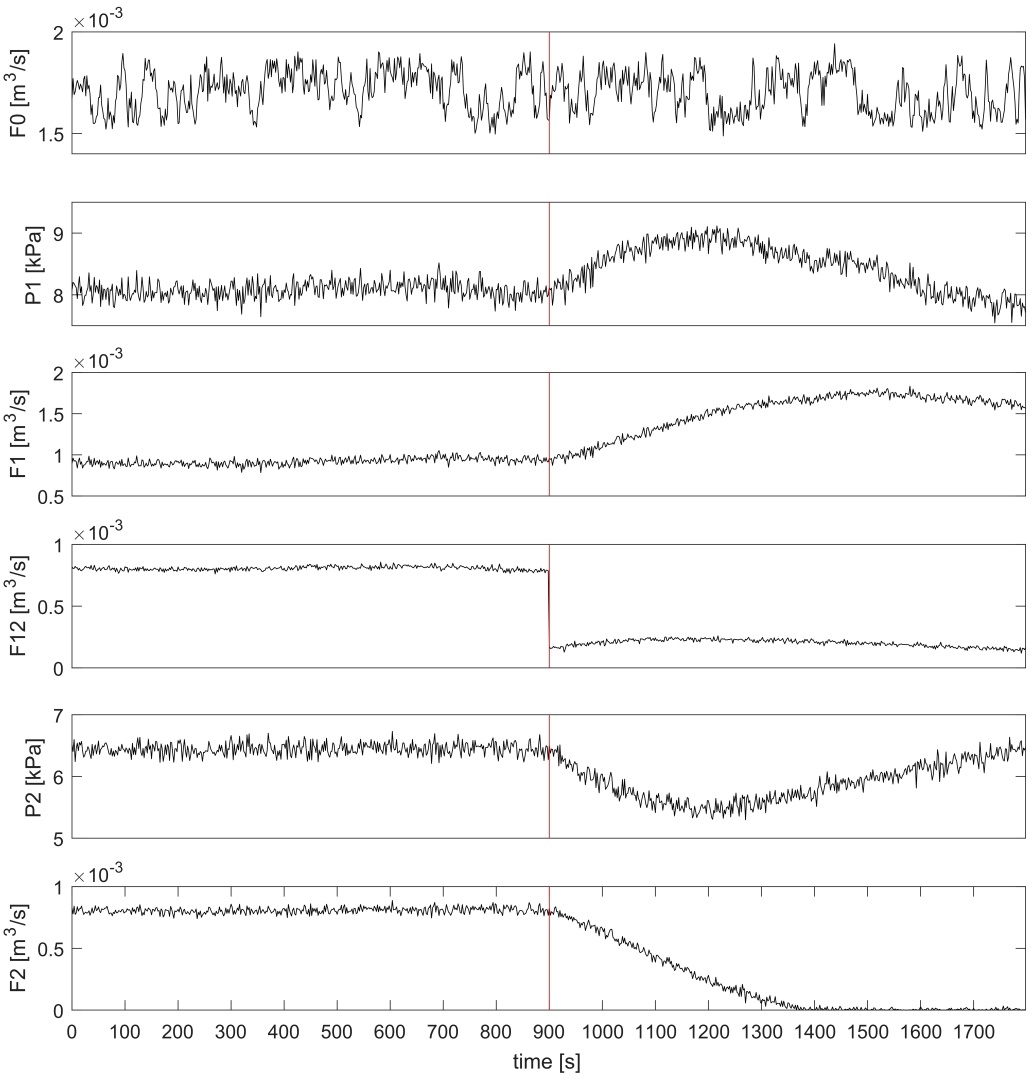


Figure C.9: Two-tank system process data: Configuration NC with aConn fault manifesting after 900 seconds.

C.3.2 iConn fault

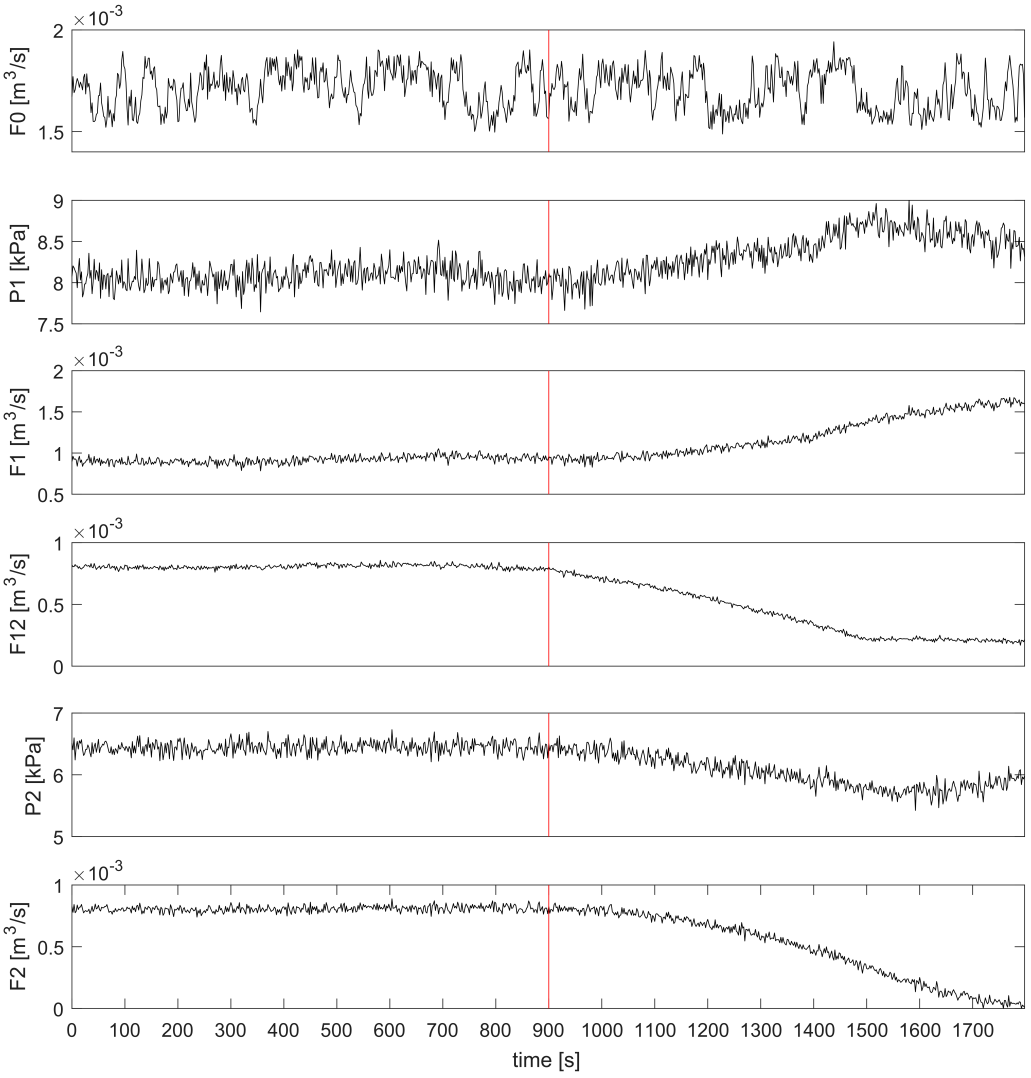


Figure C.10: Two-tank system process data: Configuration NC with iConn fault manifesting after 900 seconds.

C.3.3 P1s fault

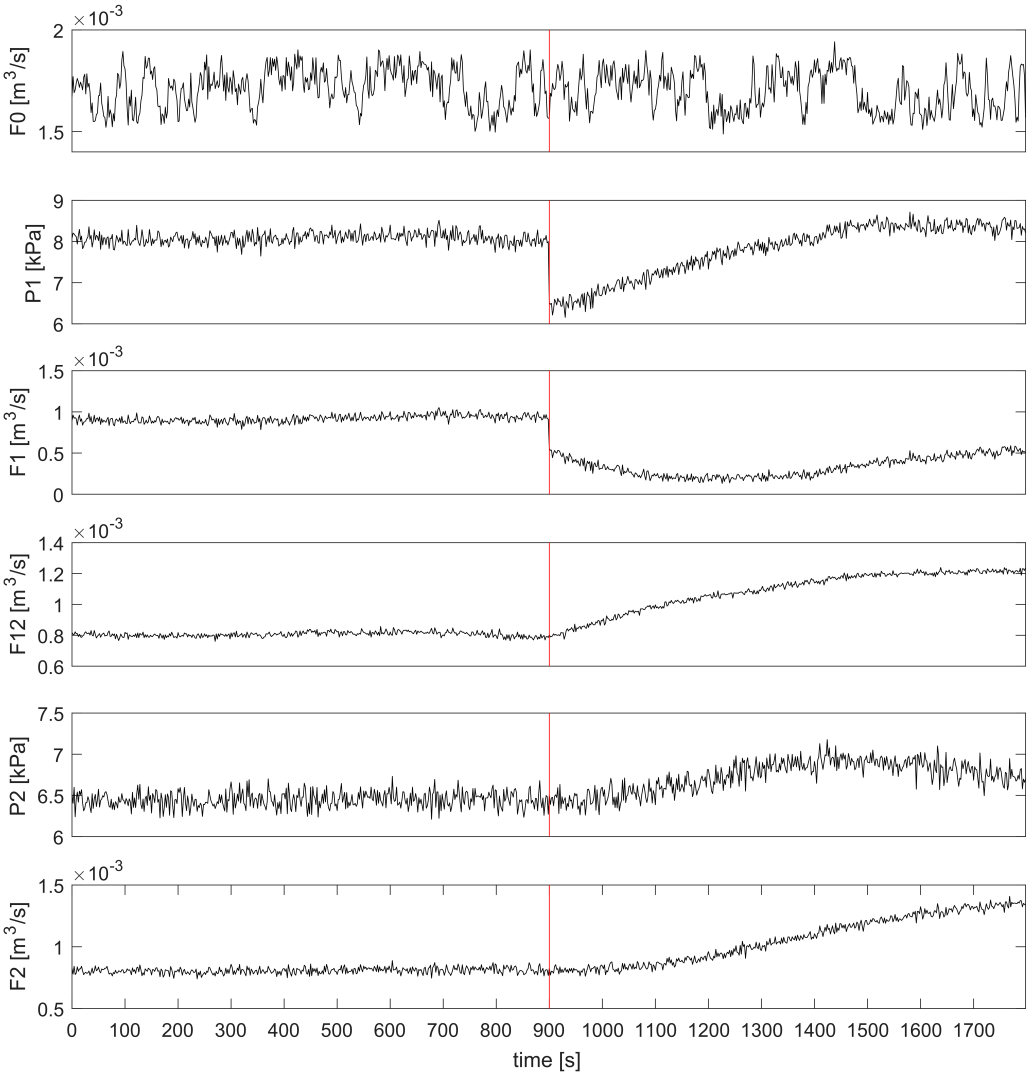


Figure C.11: Two-tank system process data: Configuration NC with P1s fault manifesting after 900 seconds.

C.3.4 P2s fault

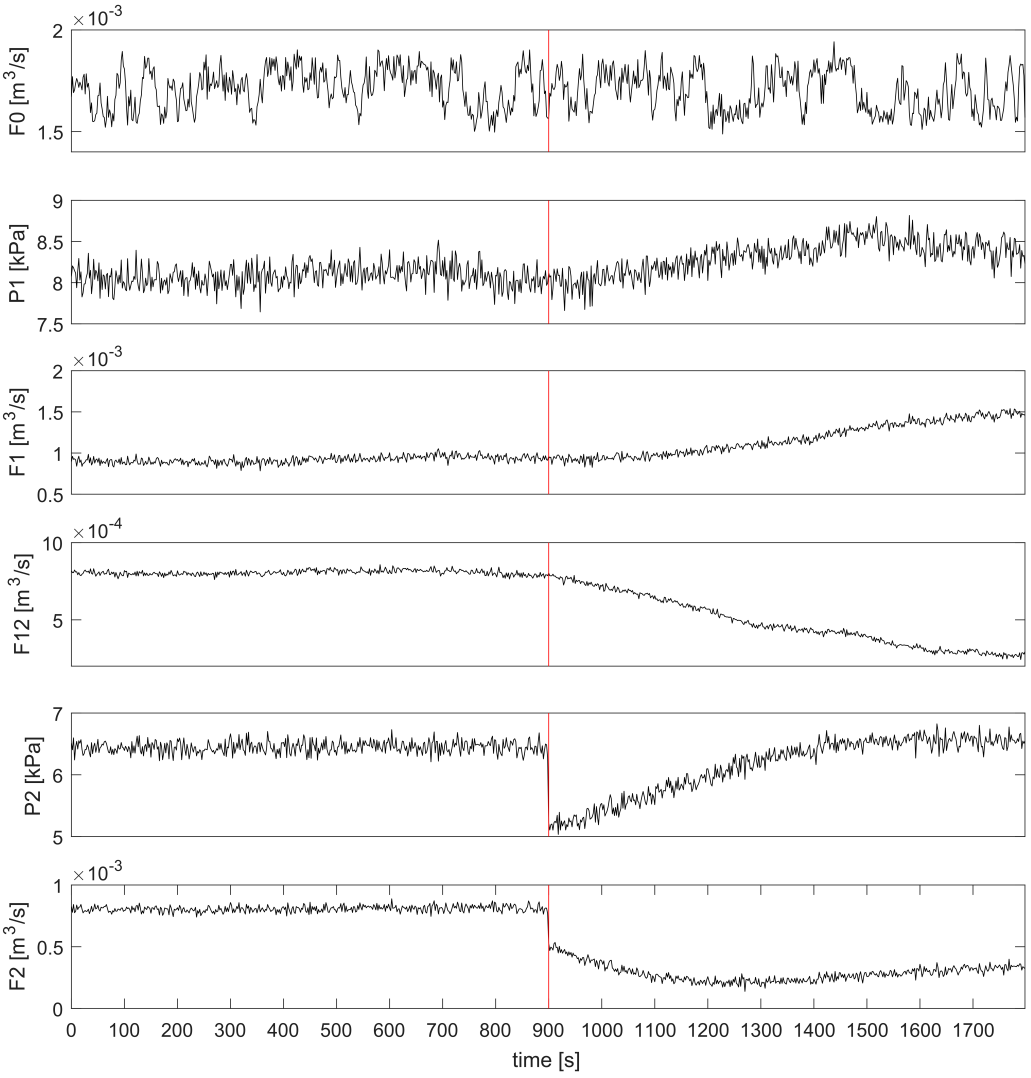


Figure C.12: Two-tank system process data: Configuration NC with P2s fault manifesting after 900 seconds.

C.4 Configuration NCR

C.4.1 aConn fault

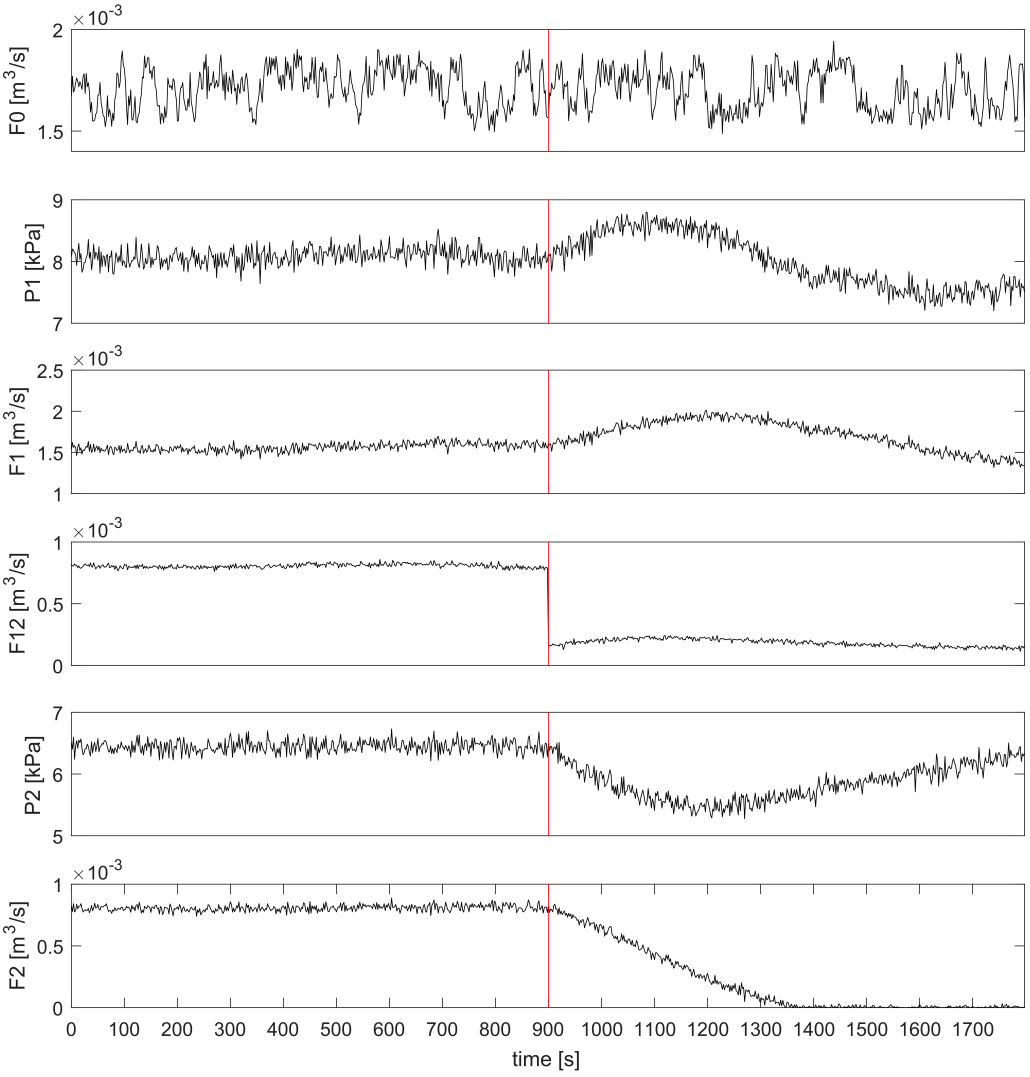


Figure C.13: Two-tank system process data: Configuration NCR with aConn fault manifesting after 900 seconds.

C.4.2 iConn fault

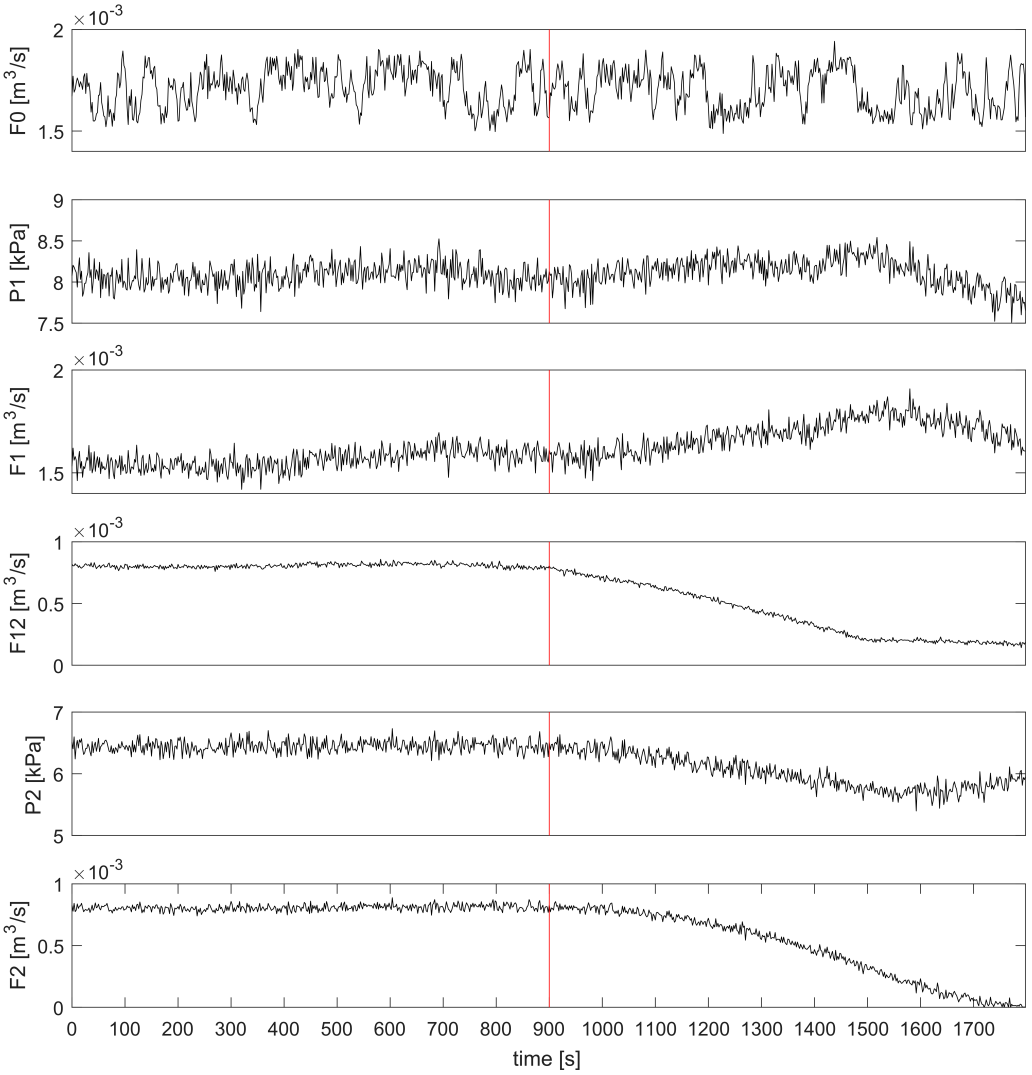


Figure C.14: Two-tank system process data: Configuration NCR with iConn fault manifesting after 900 seconds.

C.4.3 P1s fault

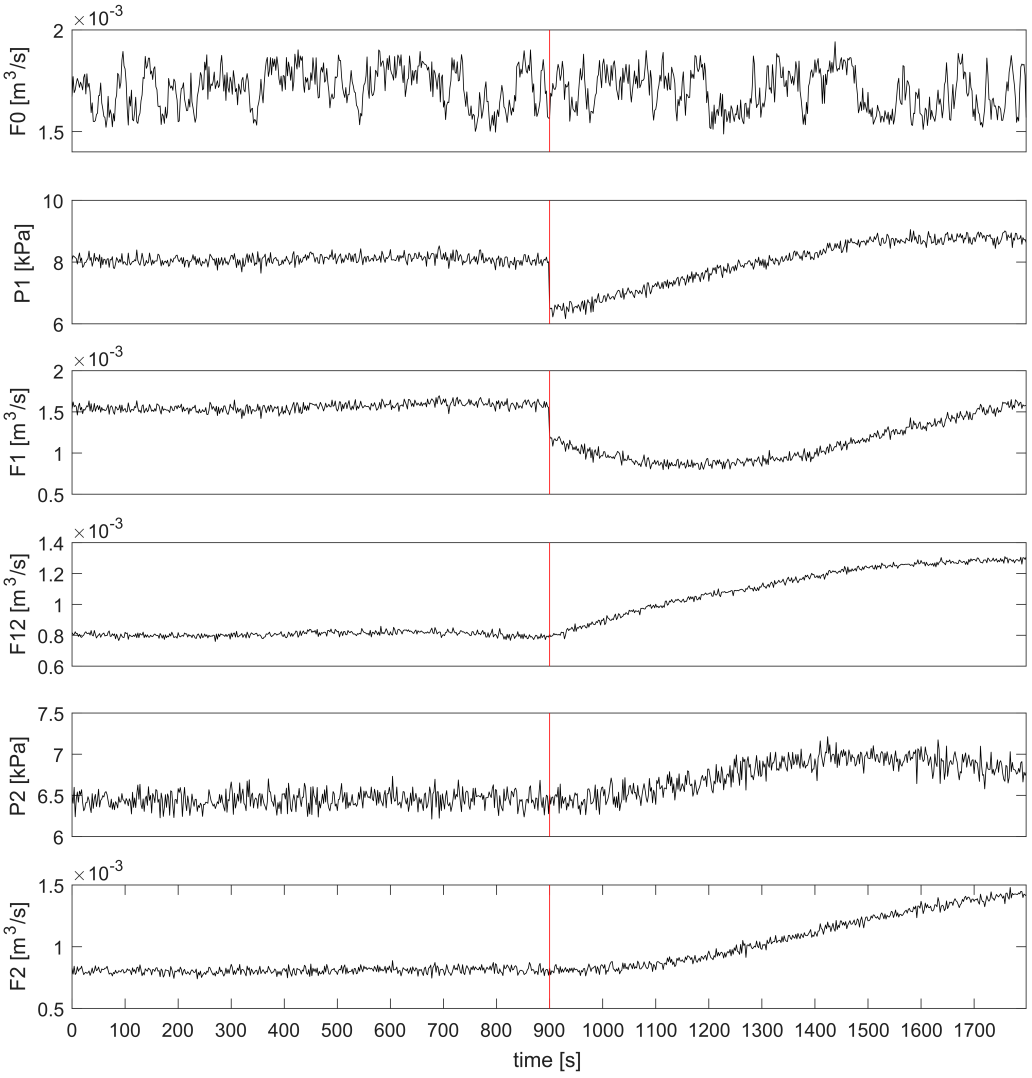


Figure C.15: Two-tank system process data: Configuration NCR with P1s fault manifesting after 900 seconds.

C.4.4 P2s fault

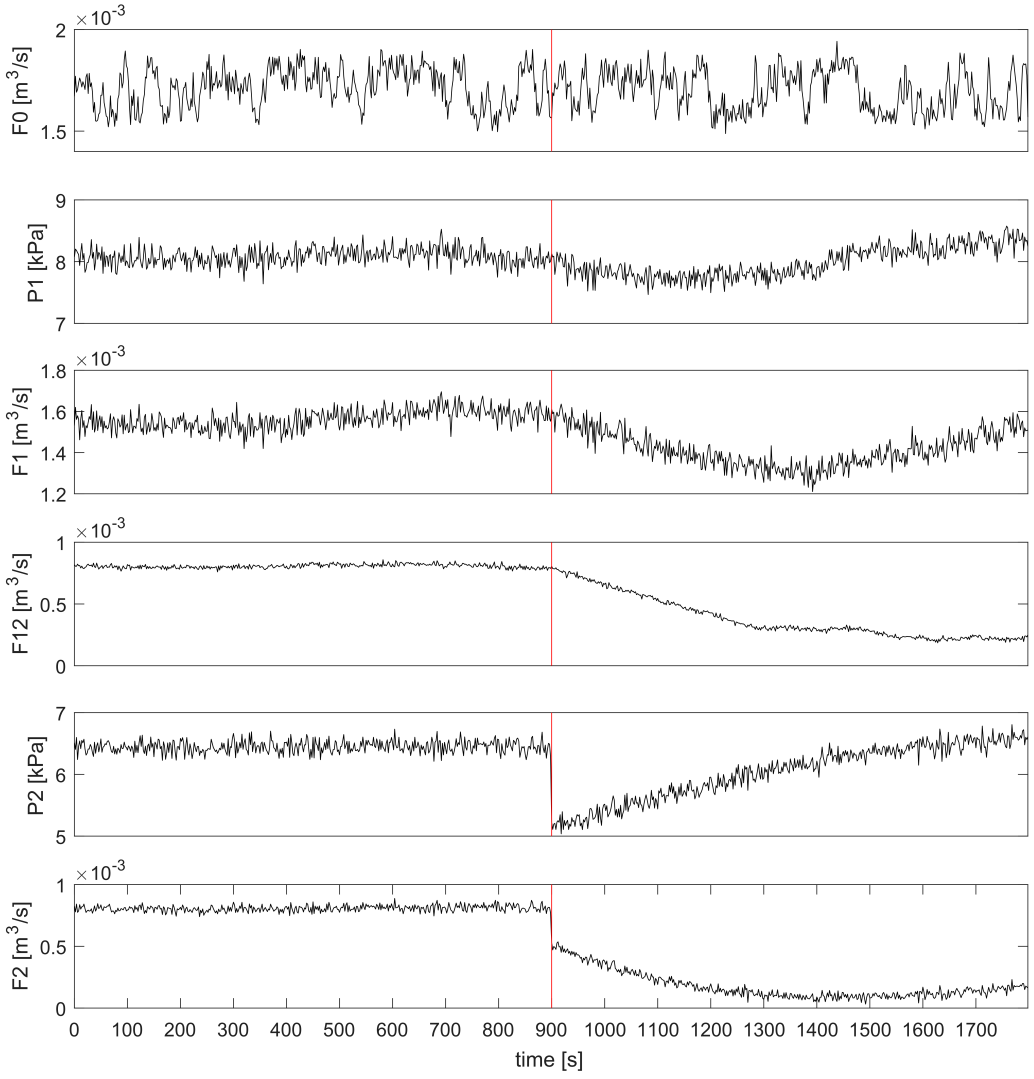


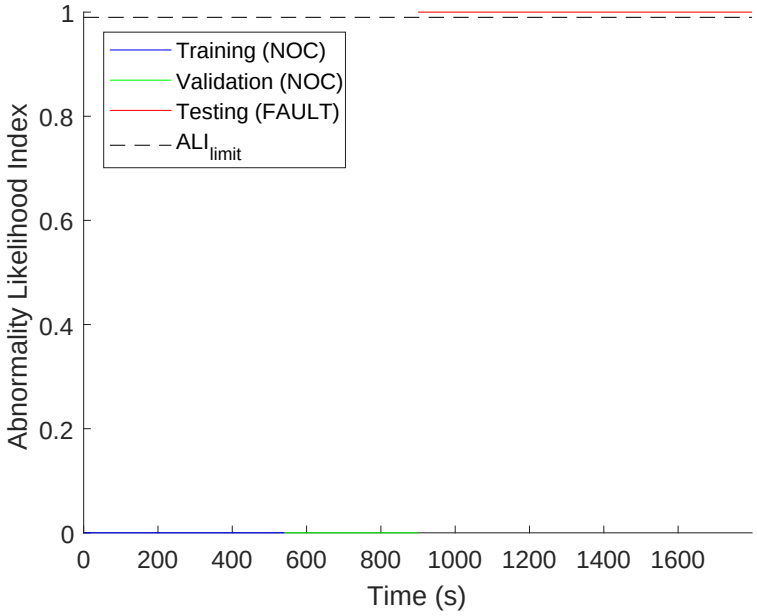
Figure C.16: Two-tank system process data: Configuration NCR with P2s fault manifesting after 900 seconds.

Appendix D

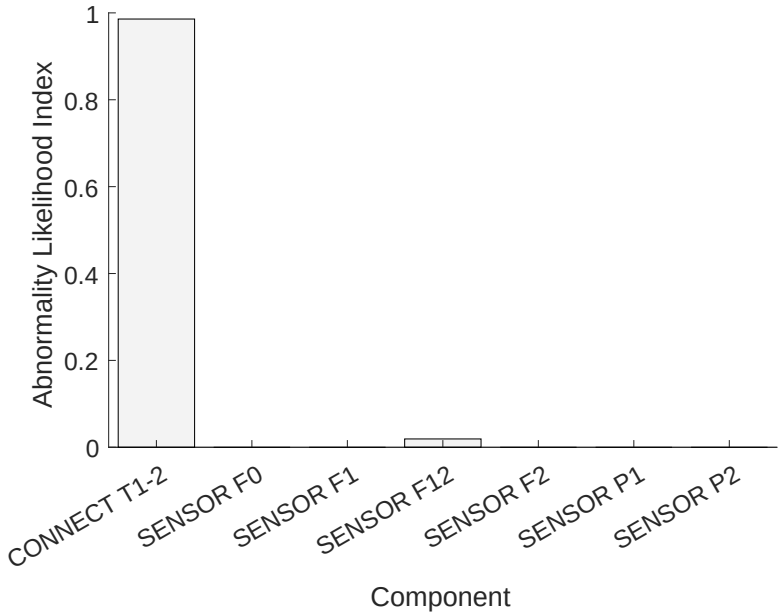
Probabilistic Fault Diagnosis: Two-tank system

D.1 Configuration sN

D.1.1 aConn fault



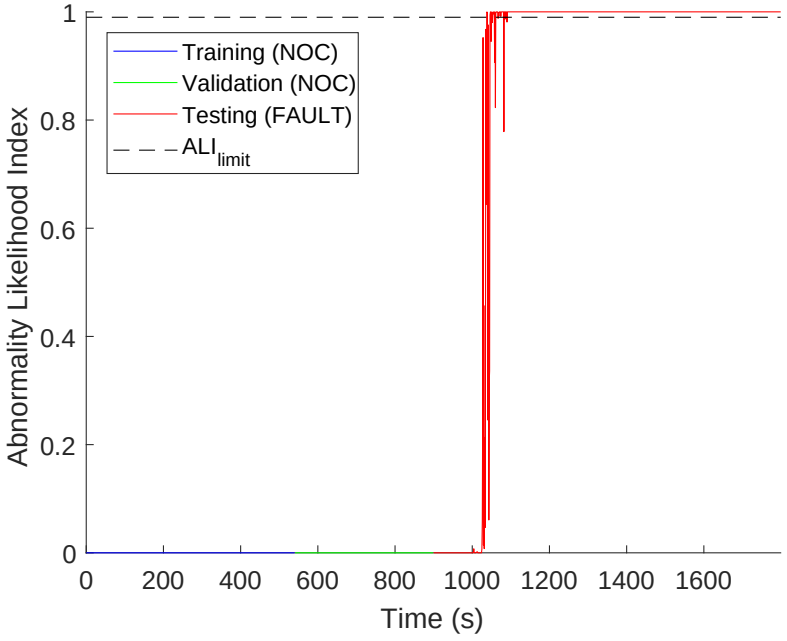
(a) Abnormality likelihood index over time.



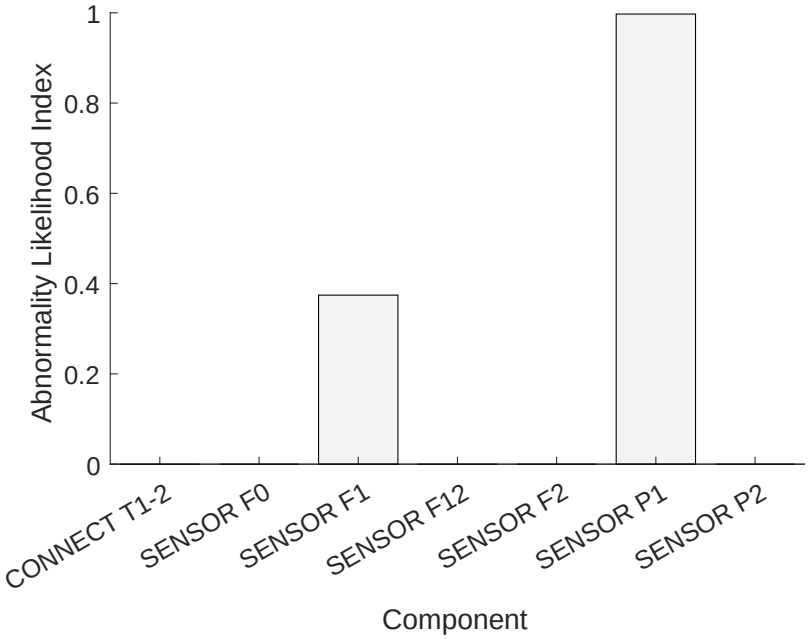
(b) Average component abnormality likelihood indexes.

Figure D.1: PD approach fault diagnosis results: Configuration sN two-tank system with aConn fault manifesting after 900 seconds.

D.1.2 iConn fault



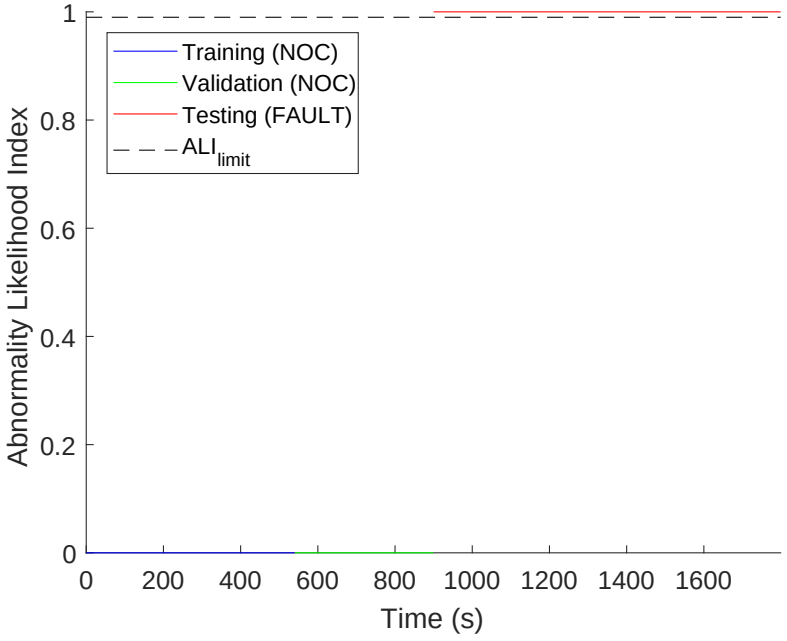
(a) Abnormality likelihood index over time.



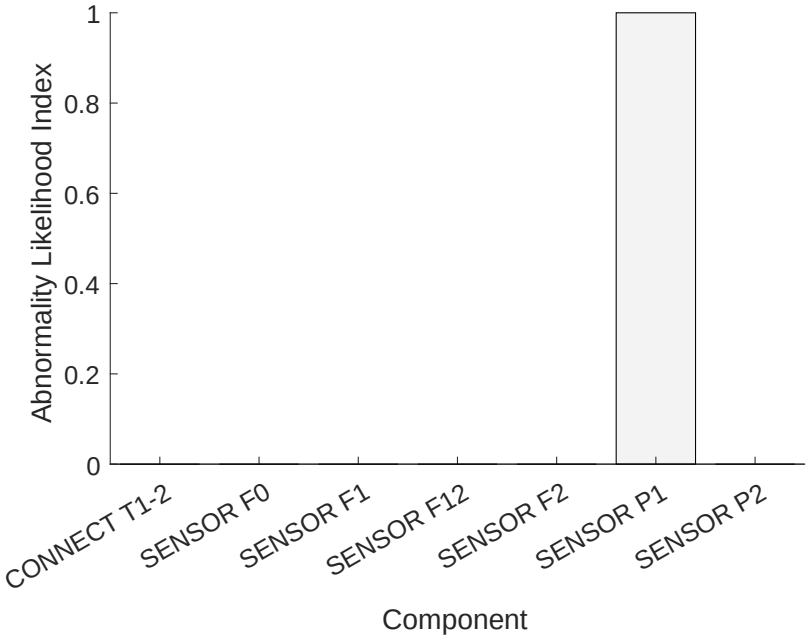
(b) Average component abnormality likelihood indexes.

Figure D.2: PD approach fault diagnosis results: Configuration sN two-tank system with iConn fault manifesting after 900 seconds.

D.1.3 P1s fault



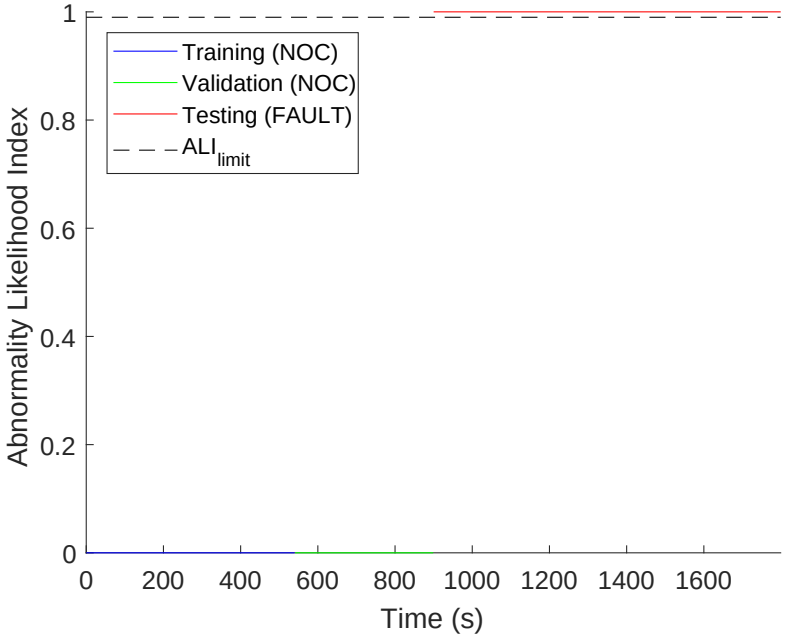
(a) Abnormality likelihood index over time.



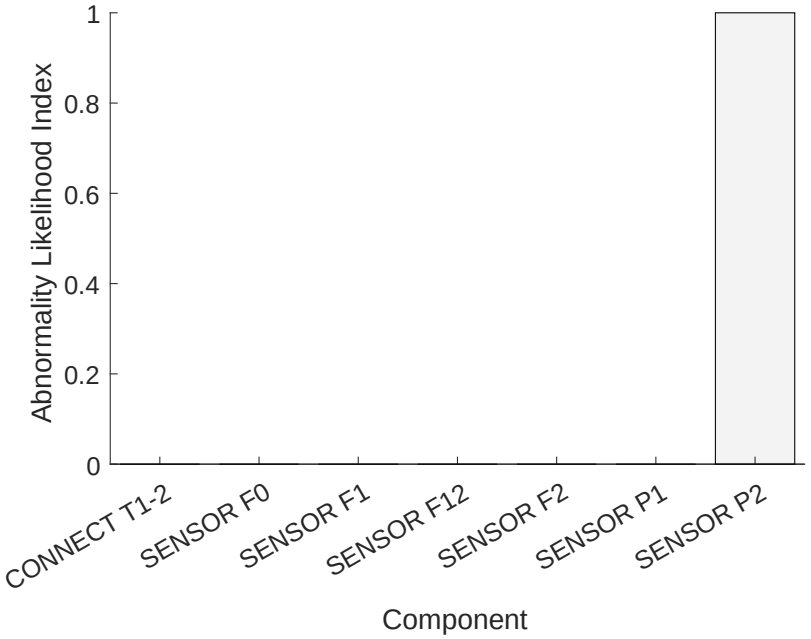
(b) Average component abnormality likelihood indexes.

Figure D.3: PD approach fault diagnosis results: Configuration sN two-tank system with P1s fault manifesting after 900 seconds.

D.1.4 P2s fault



(a) Abnormality likelihood index over time.

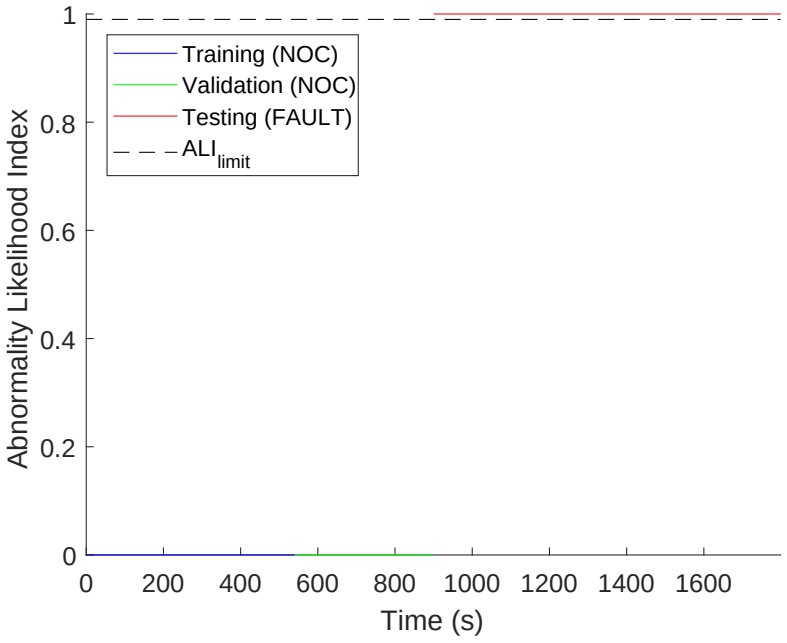


(b) Average component abnormality likelihood indexes.

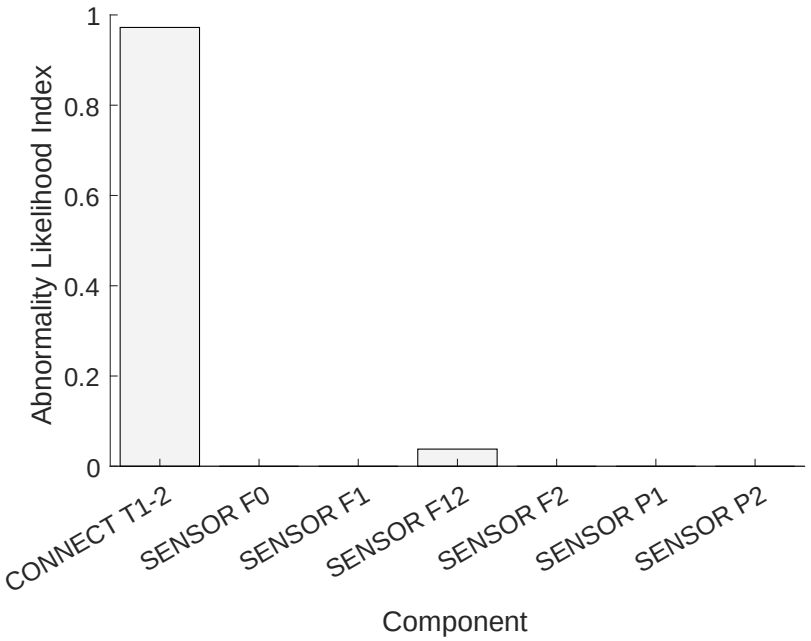
Figure D.4: PD approach fault diagnosis results: Configuration sN two-tank system with P2s fault manifesting after 900 seconds.

D.2 Configuration N

D.2.1 aConn fault



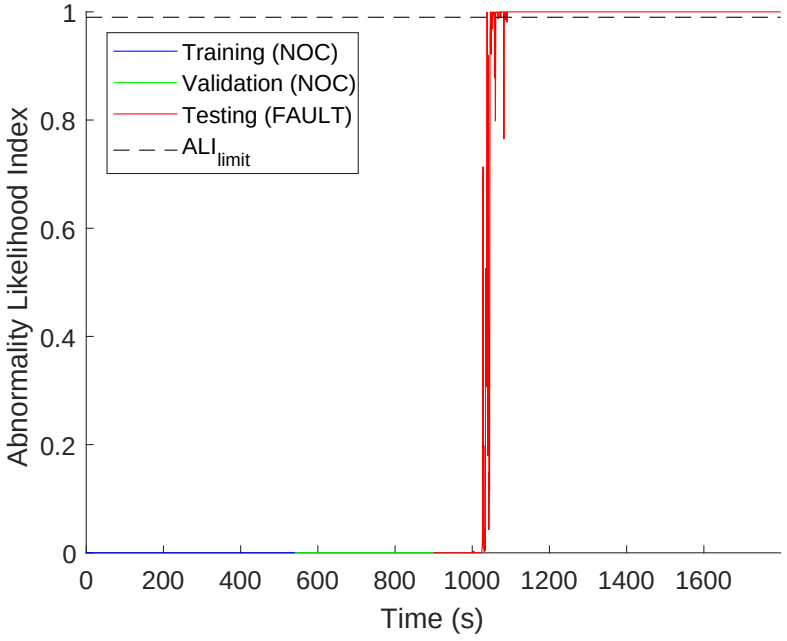
(a) Abnormality likelihood index over time.



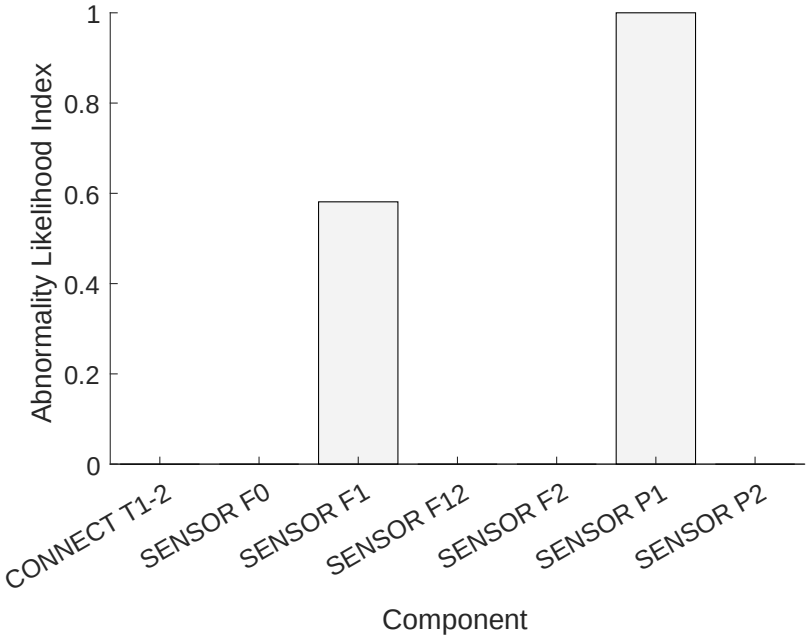
(b) Average component abnormality likelihood indexes.

Figure D.5: PD approach fault diagnosis results: Configuration N two-tank system with aConn fault manifesting after 900 seconds.

D.2.2 iConn fault



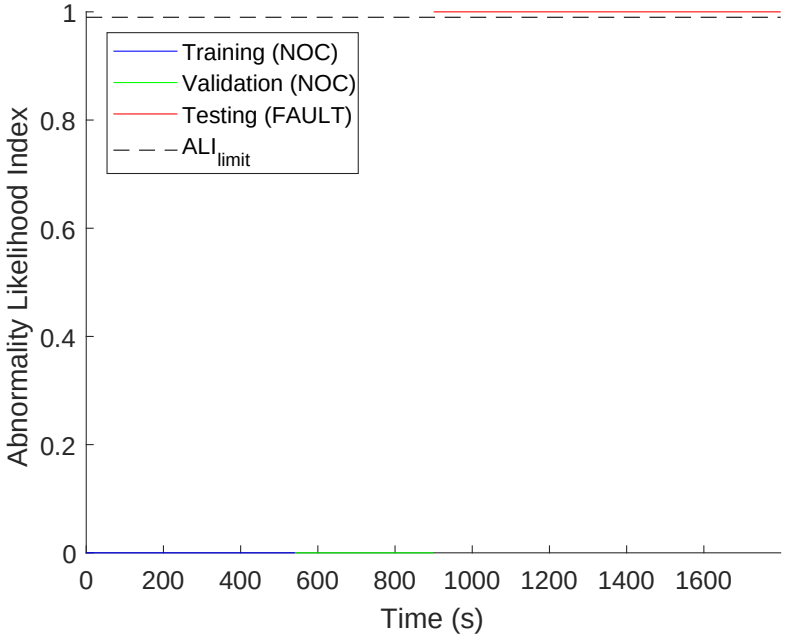
(a) Abnormality likelihood index over time.



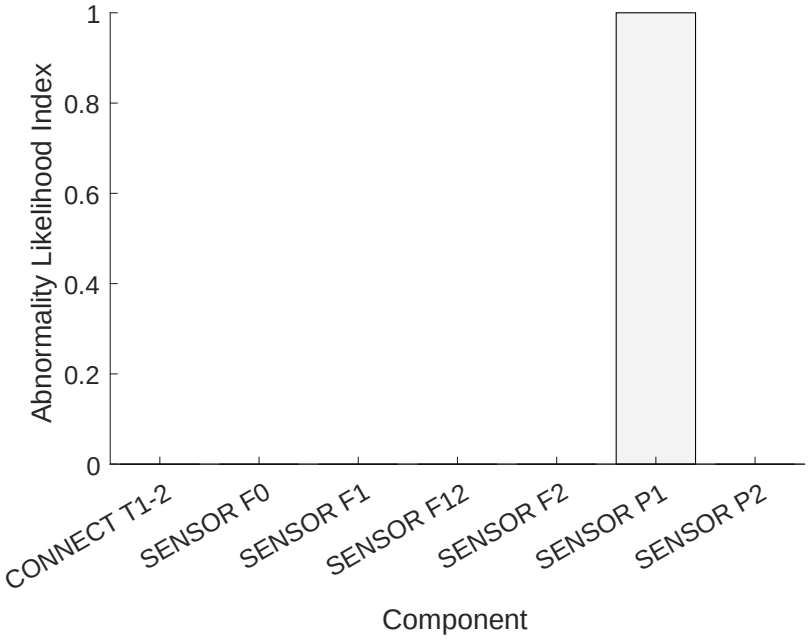
(b) Average component abnormality likelihood indexes.

Figure D.6: PD approach fault diagnosis results: Configuration N two-tank system with iConn fault manifesting after 900 seconds.

D.2.3 P1s fault



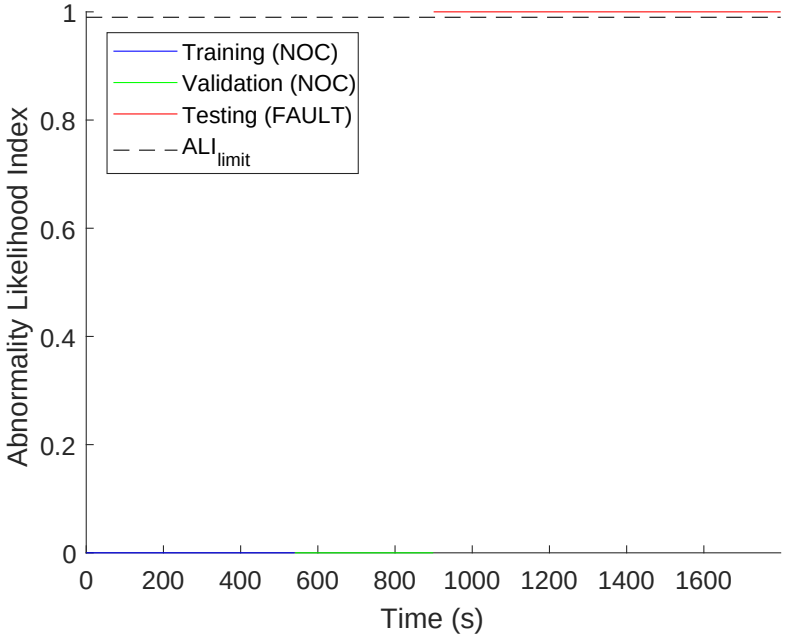
(a) Abnormality likelihood index over time.



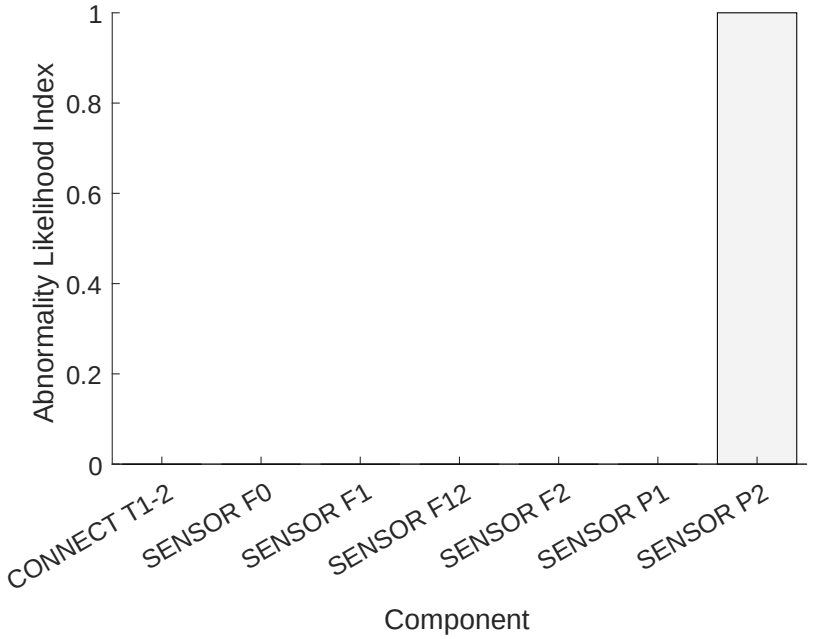
(b) Average component abnormality likelihood indexes.

Figure D.7: PD approach fault diagnosis results: Configuration N two-tank system with P1s fault manifesting after 900 seconds.

D.2.4 P2s fault



(a) Abnormality likelihood index over time.

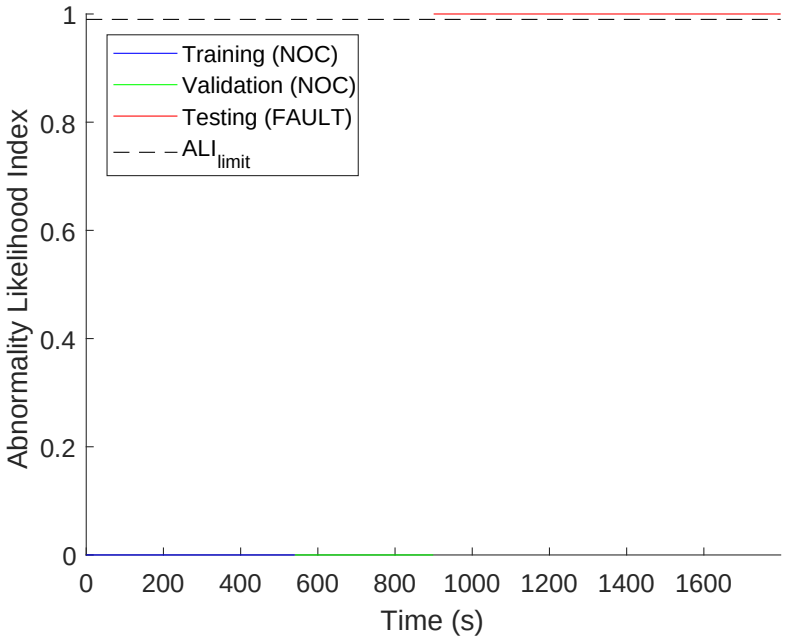


(b) Average component abnormality likelihood indexes.

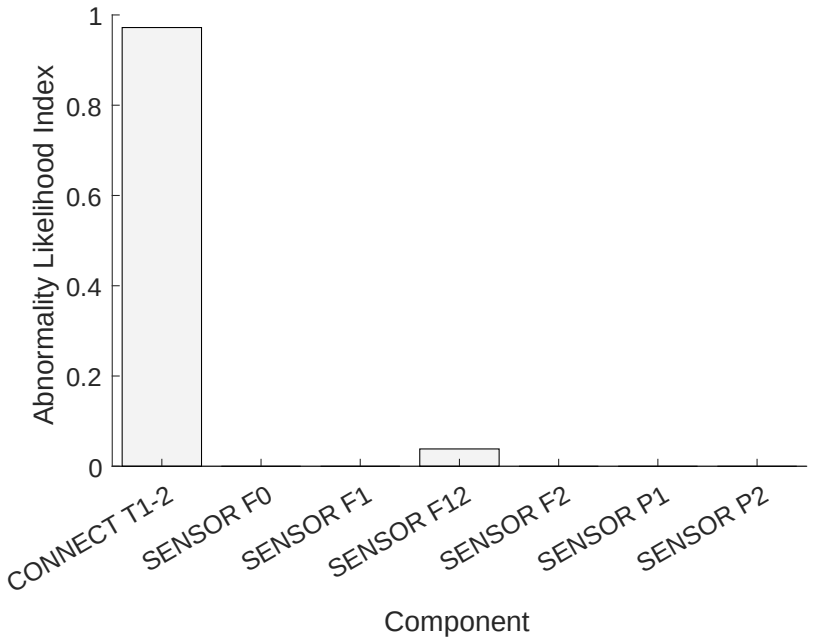
Figure D.8: PD approach fault diagnosis results: Configuration N two-tank system with P2s fault manifesting after 900 seconds.

D.3 Configuration NC

D.3.1 aConn fault



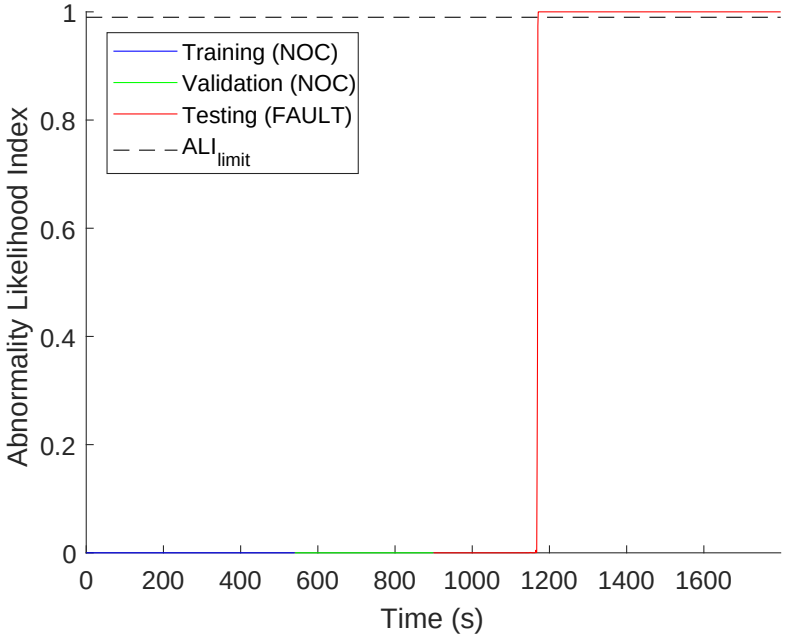
(a) Abnormality likelihood index over time.



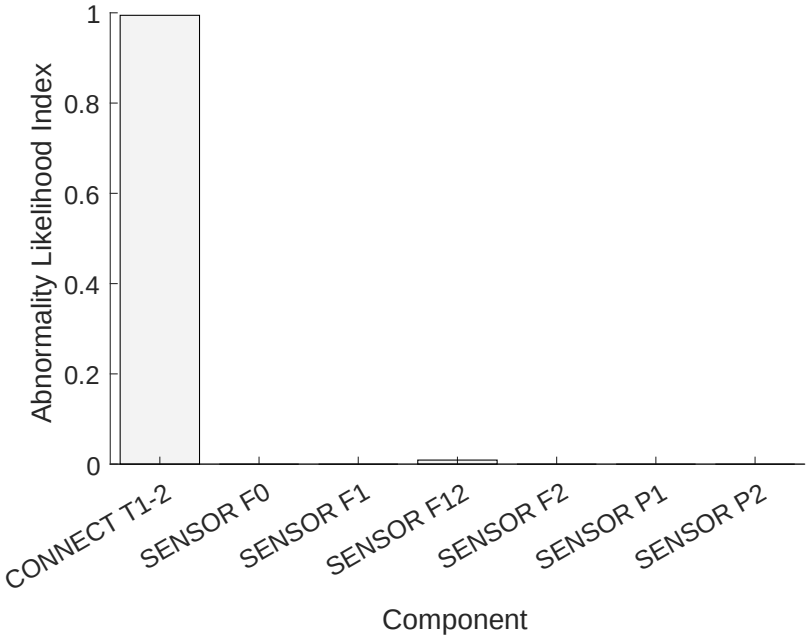
(b) Average component abnormality likelihood indexes.

Figure D.9: PD approach fault diagnosis results: Configuration NC two-tank system with aConn fault manifesting after 900 seconds.

D.3.2 iConn fault



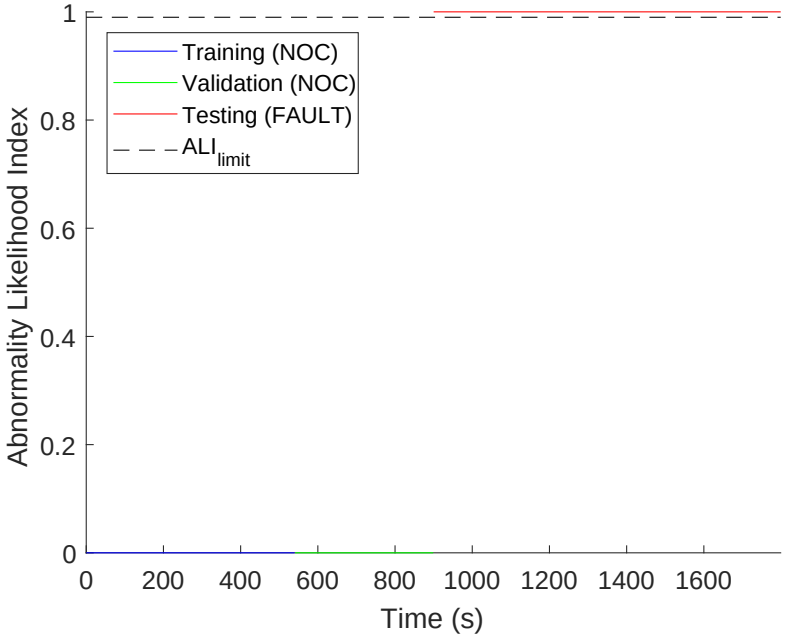
(a) Abnormality likelihood index over time.



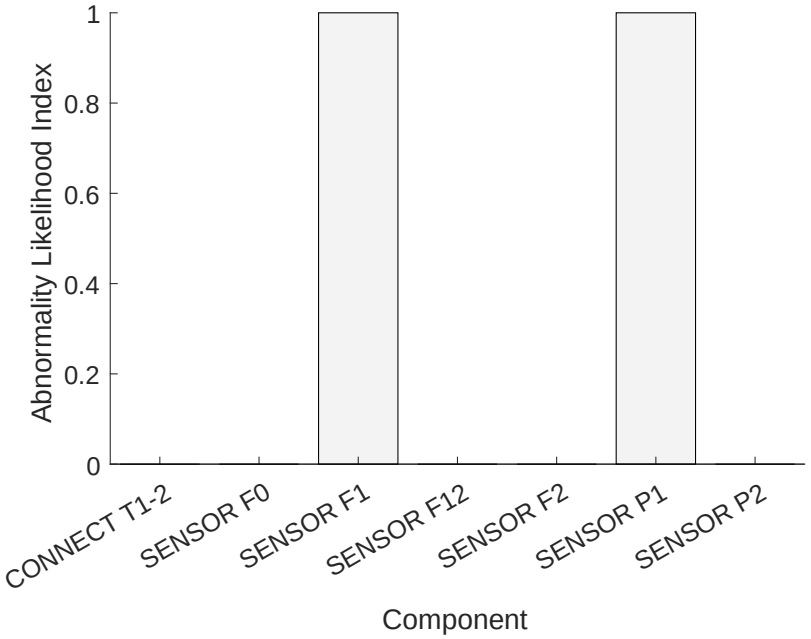
(b) Average component abnormality likelihood indexes.

Figure D.10: PD approach fault diagnosis results: Configuration NC two-tank system with iConn fault manifesting after 900 seconds.

D.3.3 P1s fault



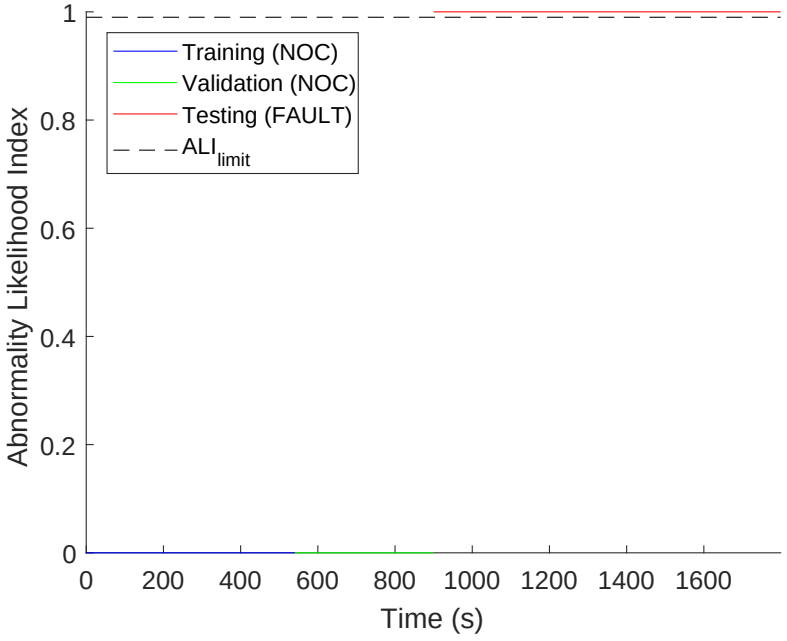
(a) Abnormality likelihood index over time.



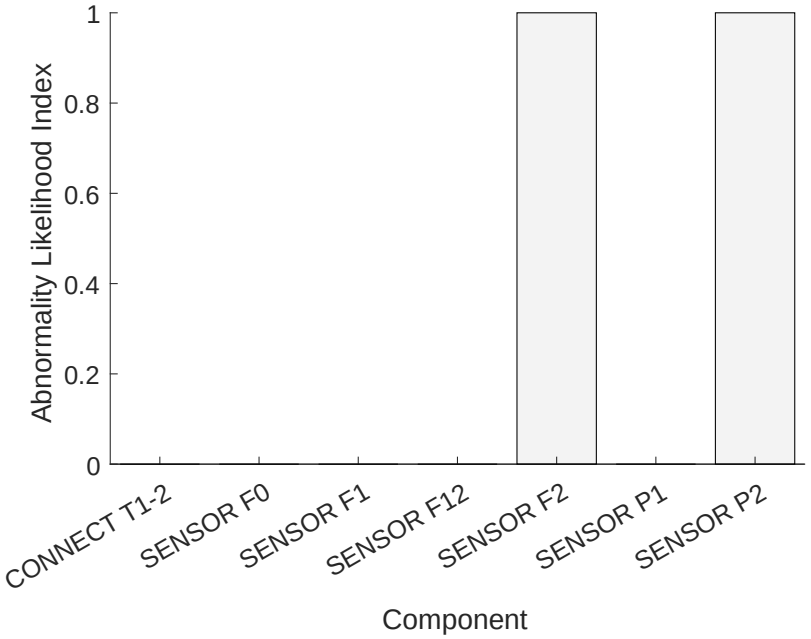
(b) Average component abnormality likelihood indexes.

Figure D.11: PD approach fault diagnosis results: Configuration NC two-tank system with P1s fault manifesting after 900 seconds.

D.3.4 P2s fault



(a) Abnormality likelihood index over time.

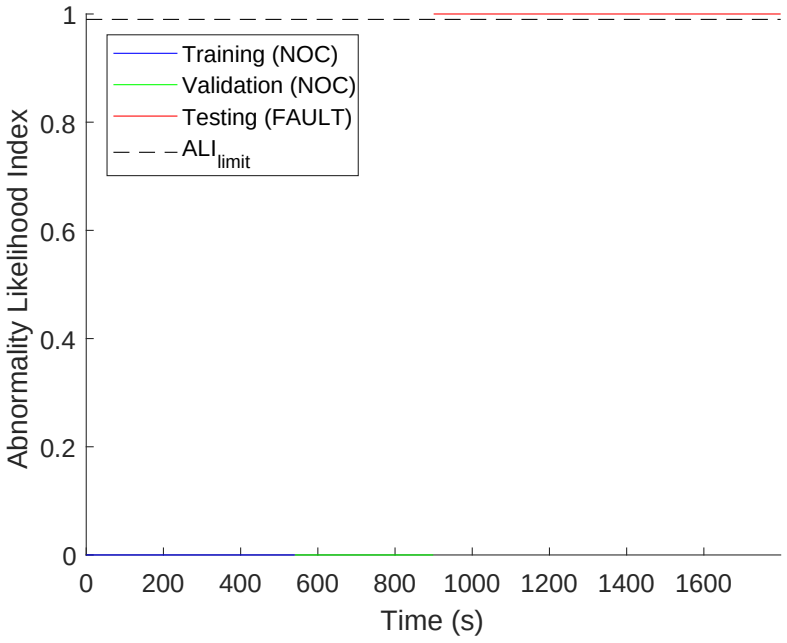


(b) Average component abnormality likelihood indexes.

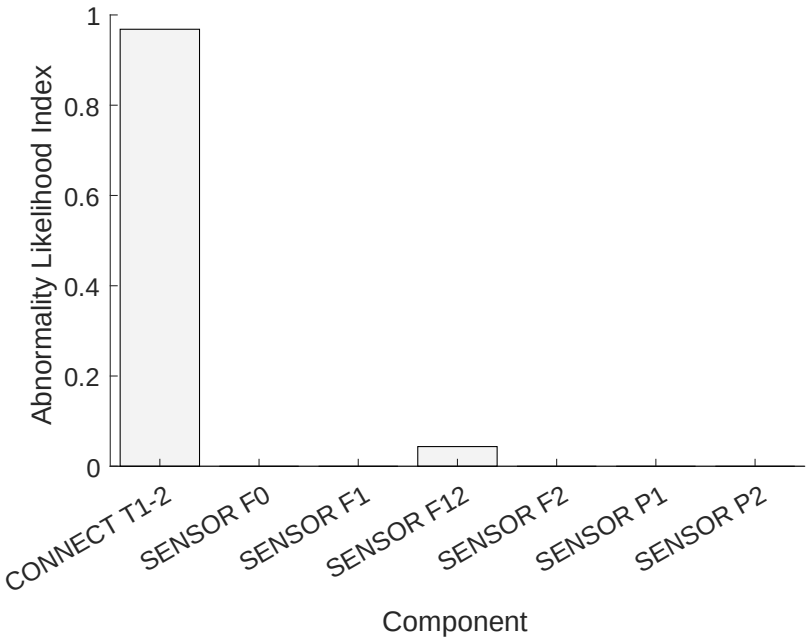
Figure D.12: PD approach fault diagnosis results: Configuration NC two-tank system with P2s fault manifesting after 900 seconds.

D.4 Configuration NCR

D.4.1 aConn fault



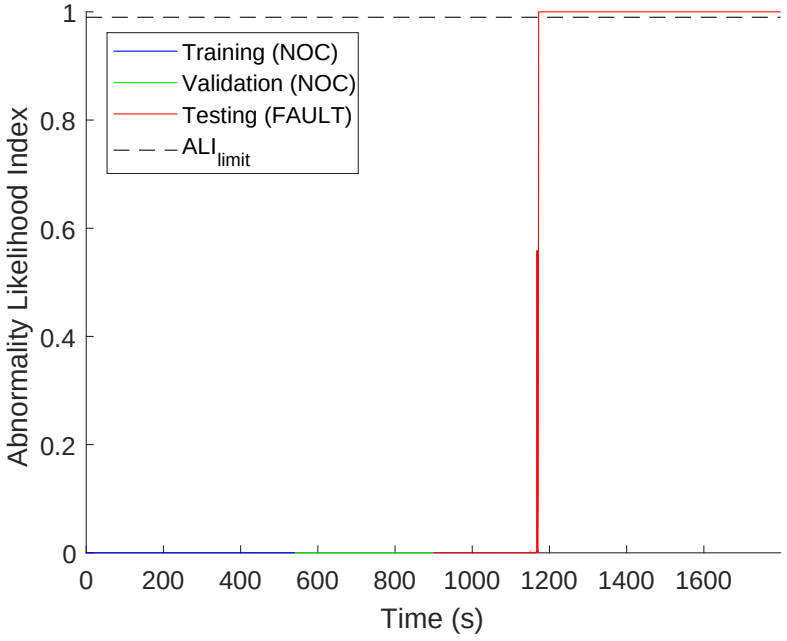
(a) Abnormality likelihood index over time.



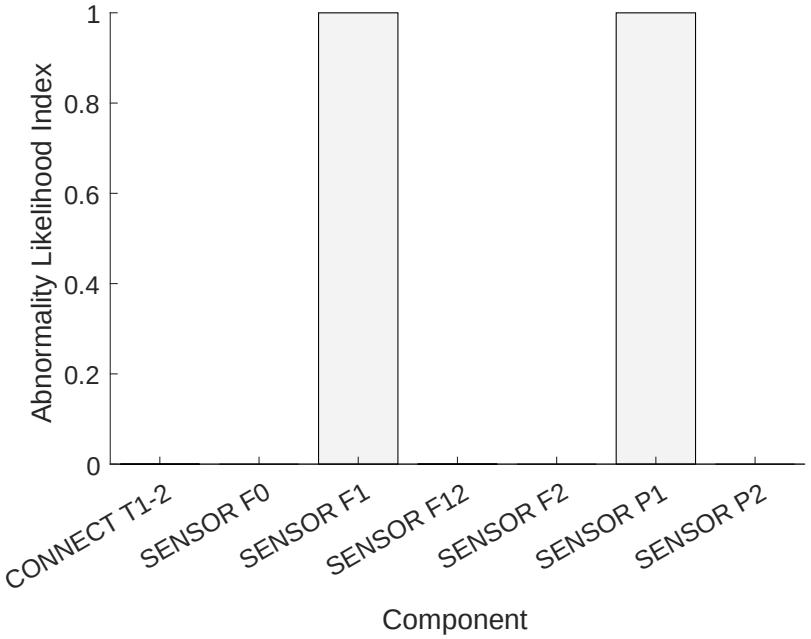
(b) Average component abnormality likelihood indexes.

Figure D.13: PD approach fault diagnosis results: Configuration NCR two-tank system with aConn fault manifesting after 900 seconds.

D.4.2 iConn fault



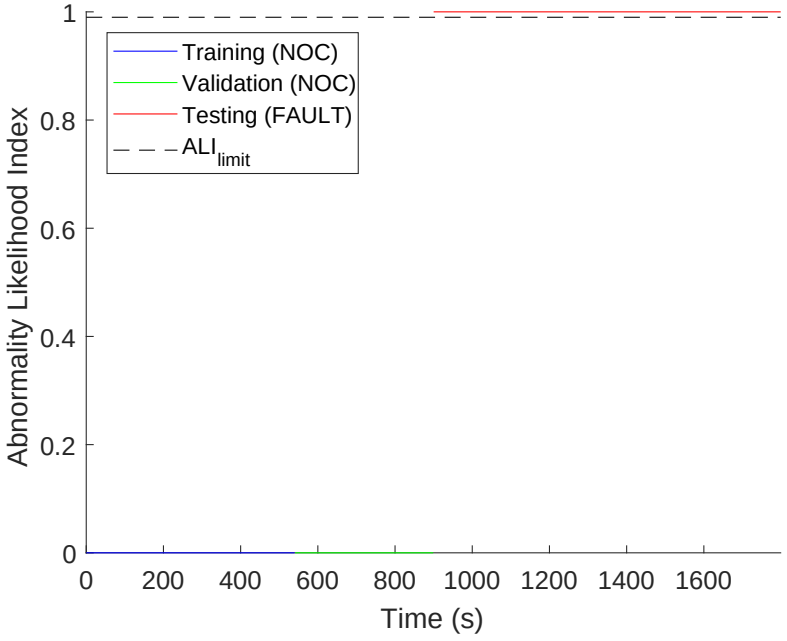
(a) Abnormality likelihood index over time.



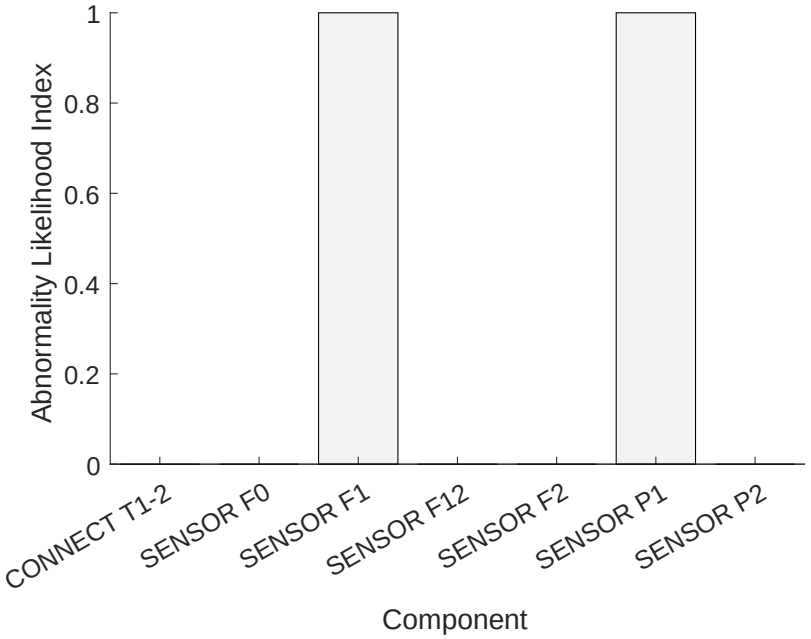
(b) Average component abnormality likelihood indexes.

Figure D.14: PD approach fault diagnosis results: Configuration NCR two-tank system with iConn fault manifesting after 900 seconds.

D.4.3 P1s fault



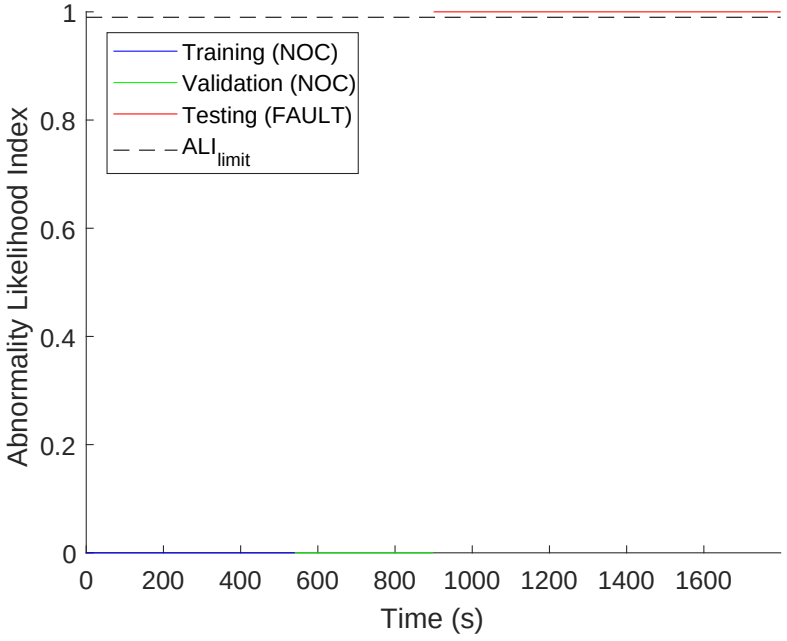
(a) Abnormality likelihood index over time.



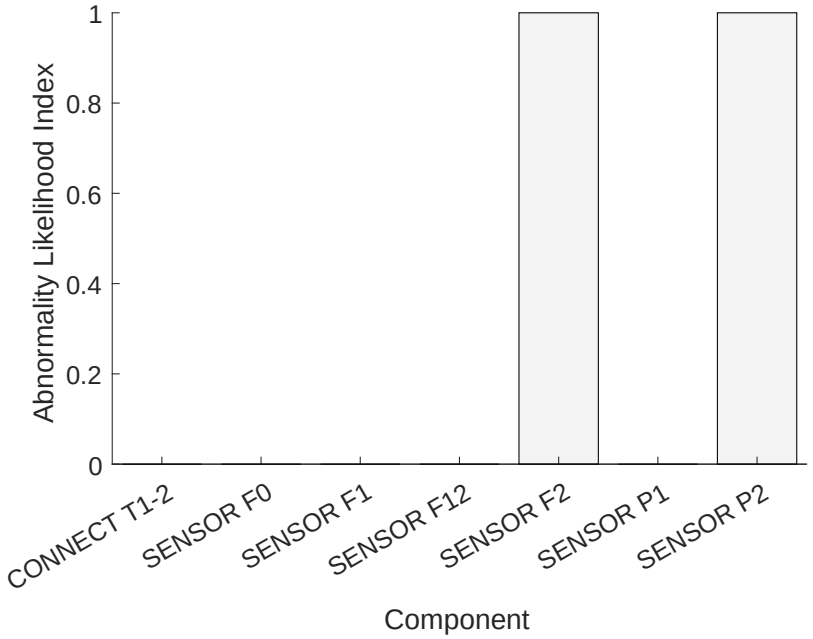
(b) Average component abnormality likelihood indexes.

Figure D.15: PD approach fault diagnosis results: Configuration NCR two-tank system with P1s fault manifesting after 900 seconds.

D.4.4 P2s fault



(a) Abnormality likelihood index over time.



(b) Average component abnormality likelihood indexes.

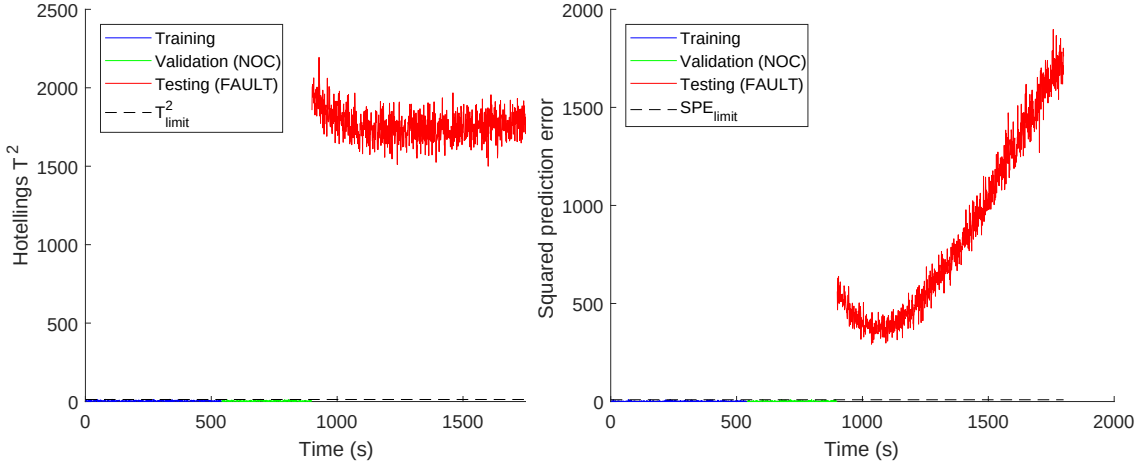
Figure D.16: PD approach fault diagnosis results: Configuration NCR two-tank system with P2s fault manifesting after 900 seconds.

Appendix E

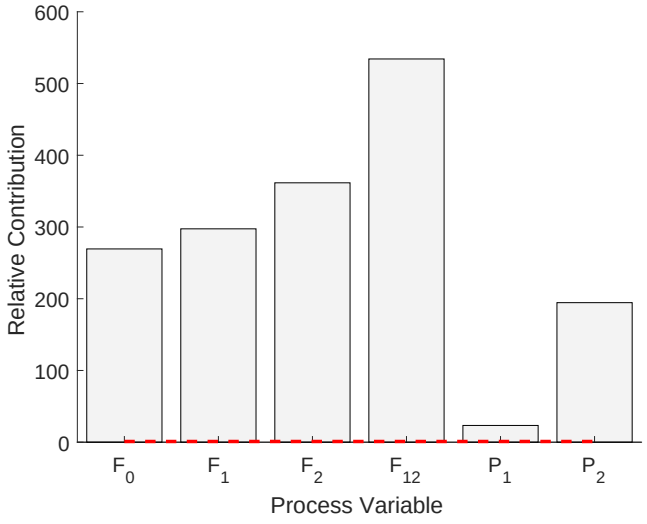
Multivariate Statistical Process Monitoring Fault Diagnosis: Two-tank system

E.1 Configuration sN

E.1.1 aConn fault



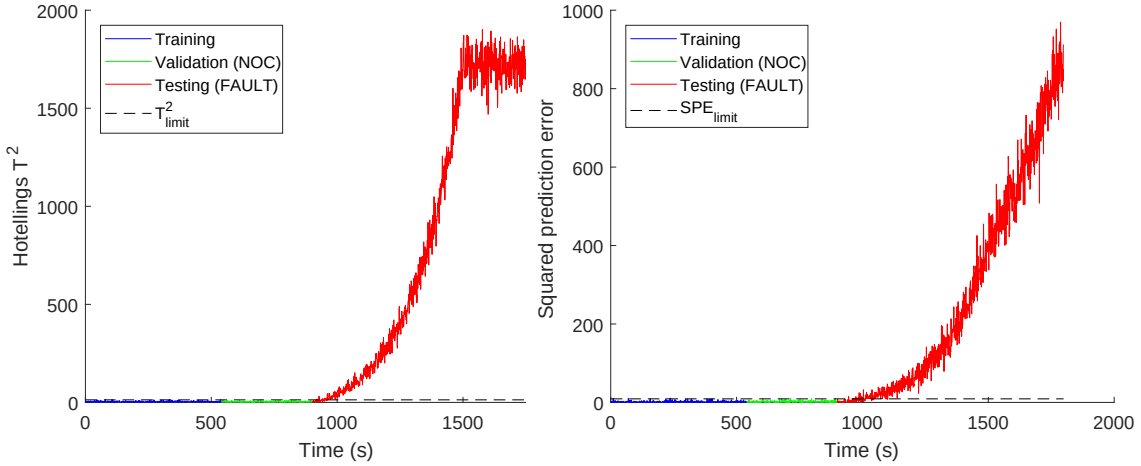
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



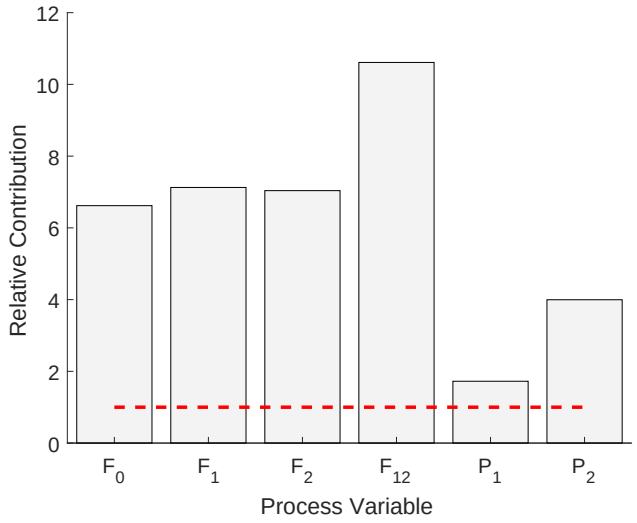
(c) Average relative contributions.

Figure E.1: MSPM approach fault diagnosis results: Configuration sN two-tank system with aConn fault manifesting after 900 seconds.

E.1.2 iConn fault



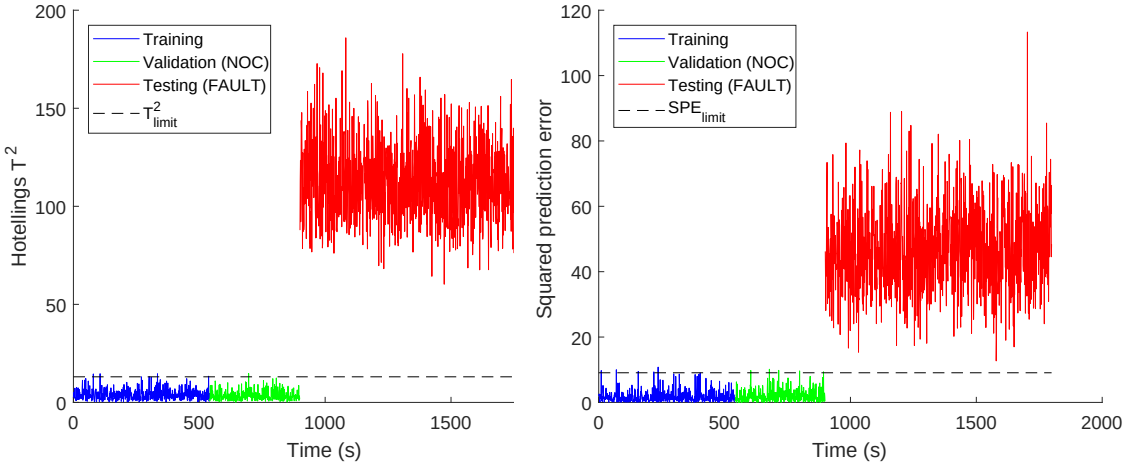
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



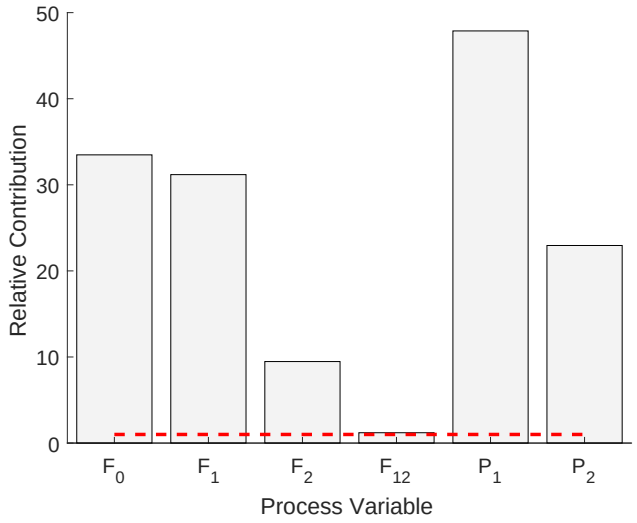
(c) Average relative contributions.

Figure E.2: MSPM approach fault diagnosis results: Configuration sN two-tank system with iConn fault manifesting after 900 seconds.

E.1.3 P1s fault



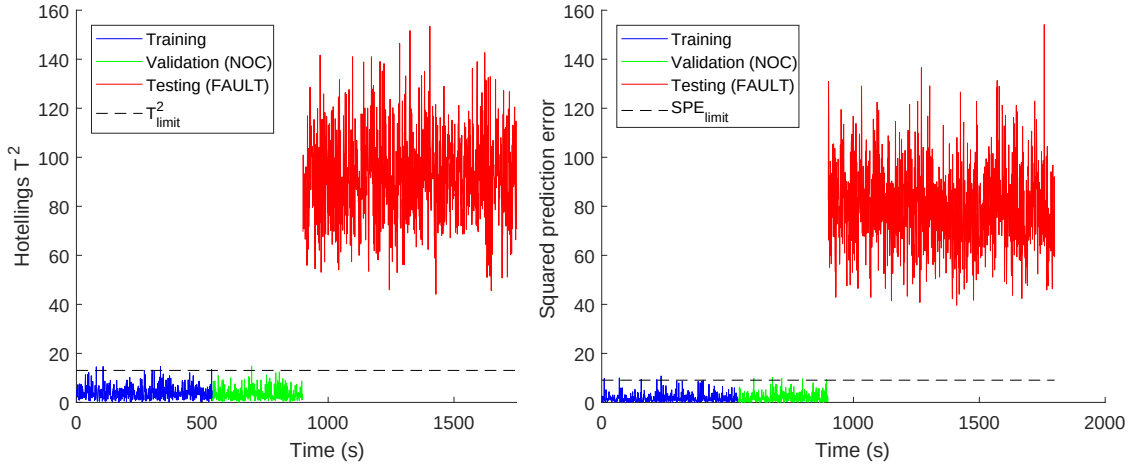
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



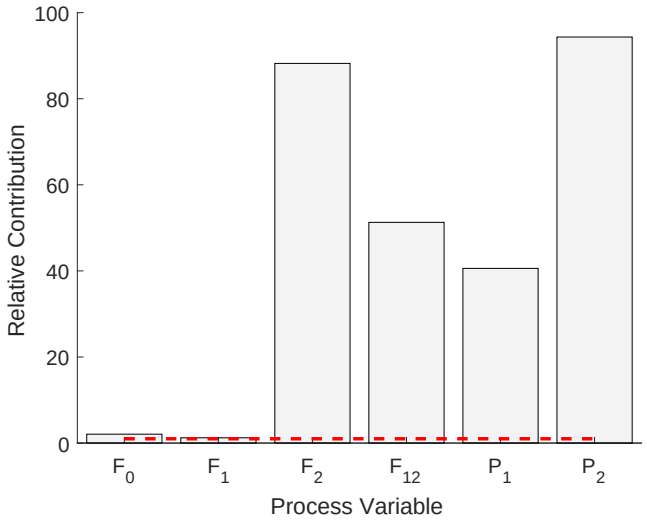
(c) Average relative contributions.

Figure E.3: MSPM approach fault diagnosis results: Configuration sN two-tank system with P1s fault manifesting after 900 seconds.

E.1.4 P2s fault



(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.

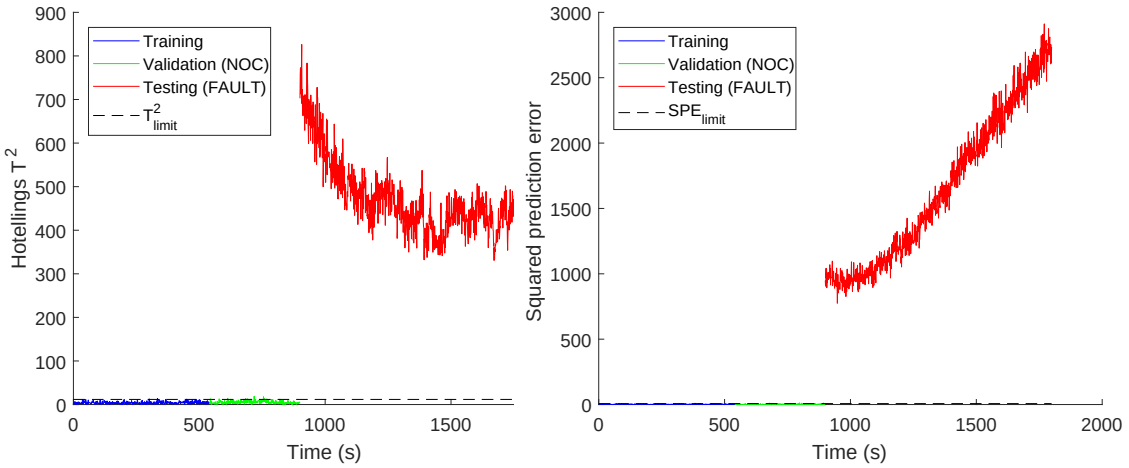


(c) Average relative contributions.

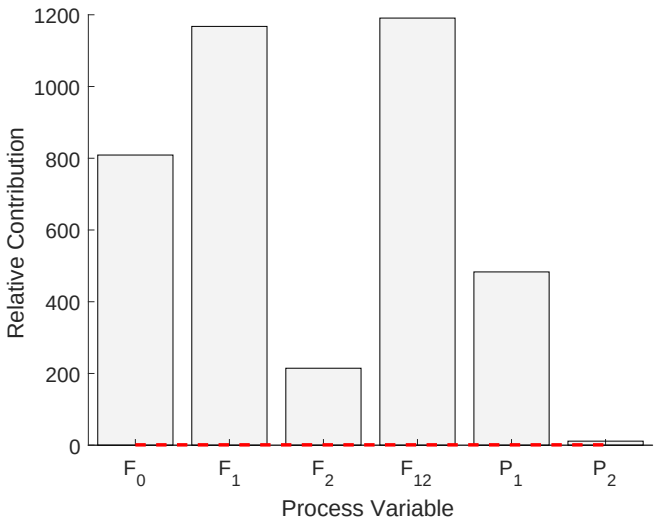
Figure E.4: MSPM approach fault diagnosis results: Configuration sN two-tank system with P2s fault manifesting after 900 seconds.

E.2 Configuration N

E.2.1 aConn fault



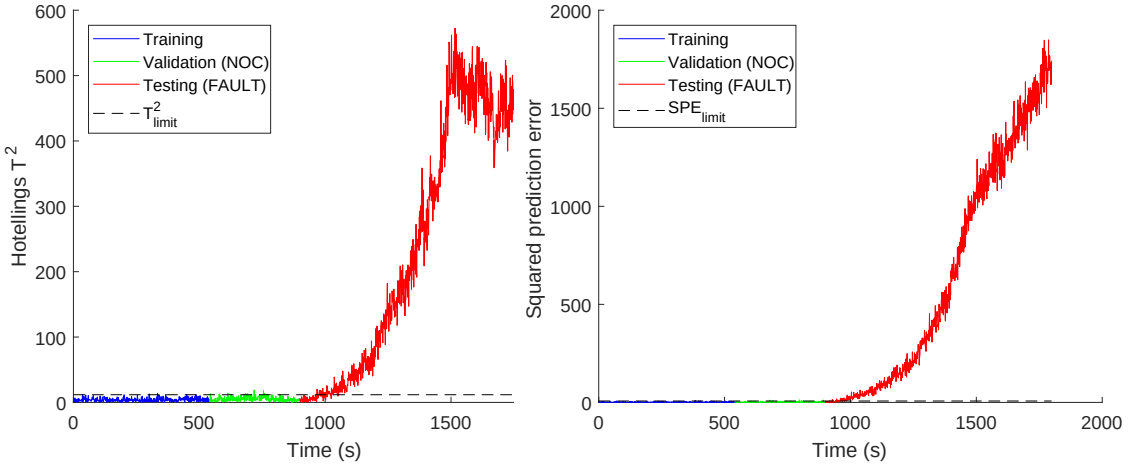
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



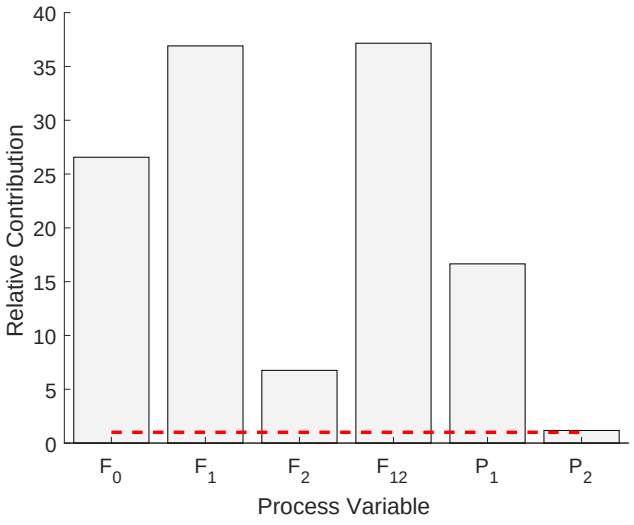
(c) Average relative contributions.

Figure E.5: MSPM approach fault diagnosis results: Configuration N two-tank system with aConn fault manifesting after 900 seconds.

E.2.2 iConn fault



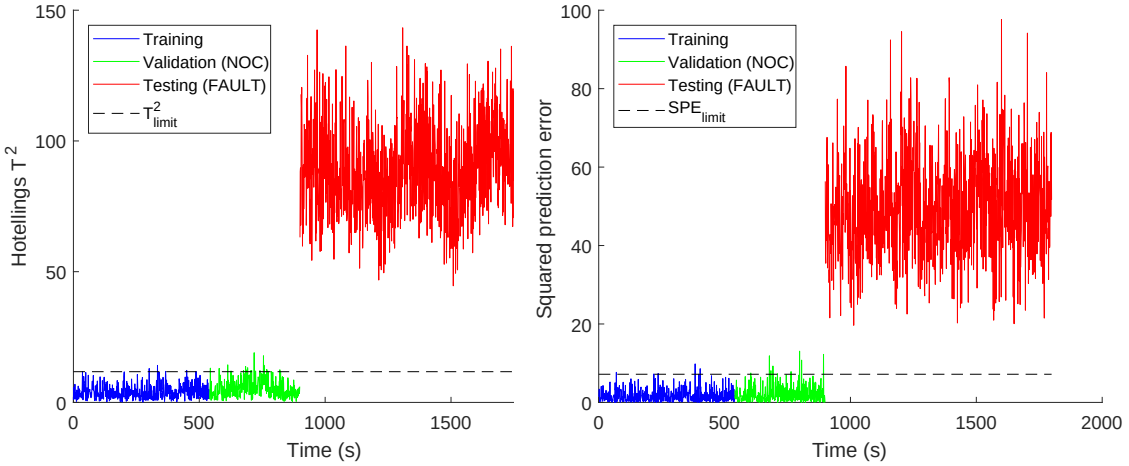
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



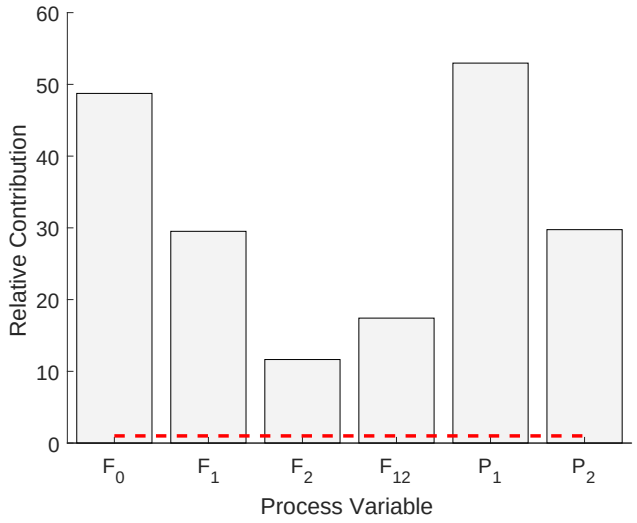
(c) Average relative contributions.

Figure E.6: MSPM approach fault diagnosis results: Configuration N two-tank system with iConn fault manifesting after 900 seconds.

E.2.3 P1s fault



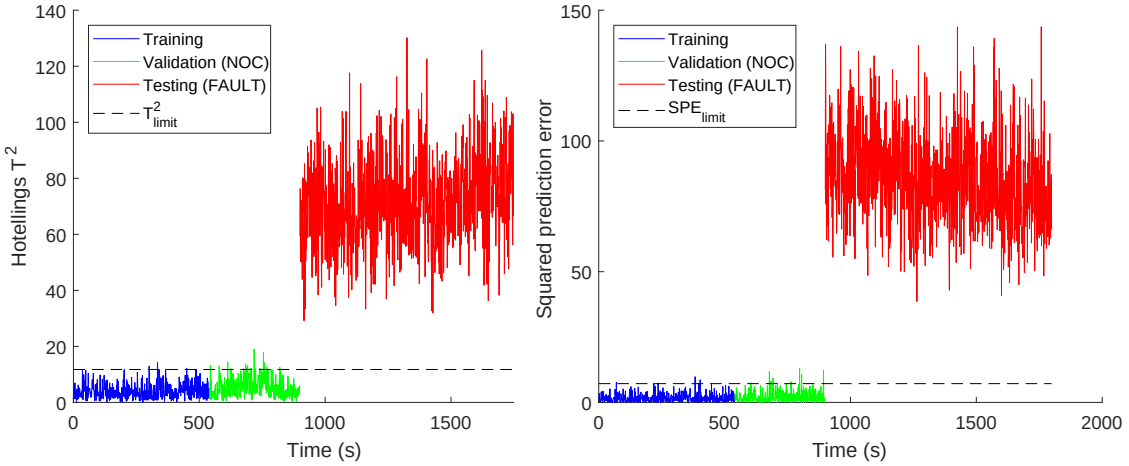
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



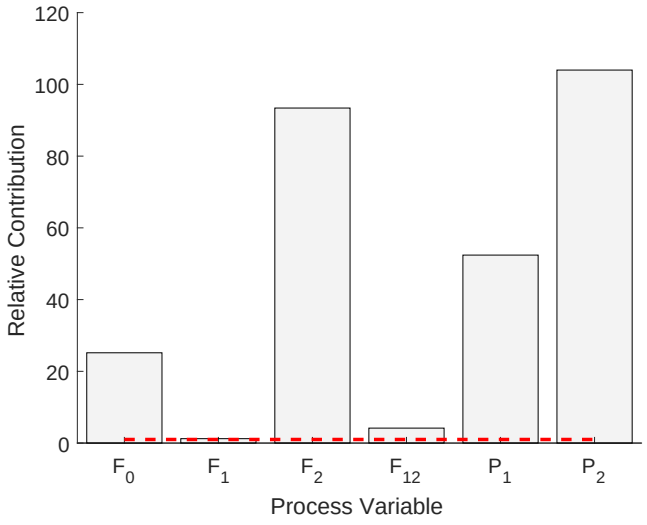
(c) Average relative contributions.

Figure E.7: MSPM approach fault diagnosis results: Configuration N two-tank system with P1s fault manifesting after 900 seconds.

E.2.4 P2s fault



(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.

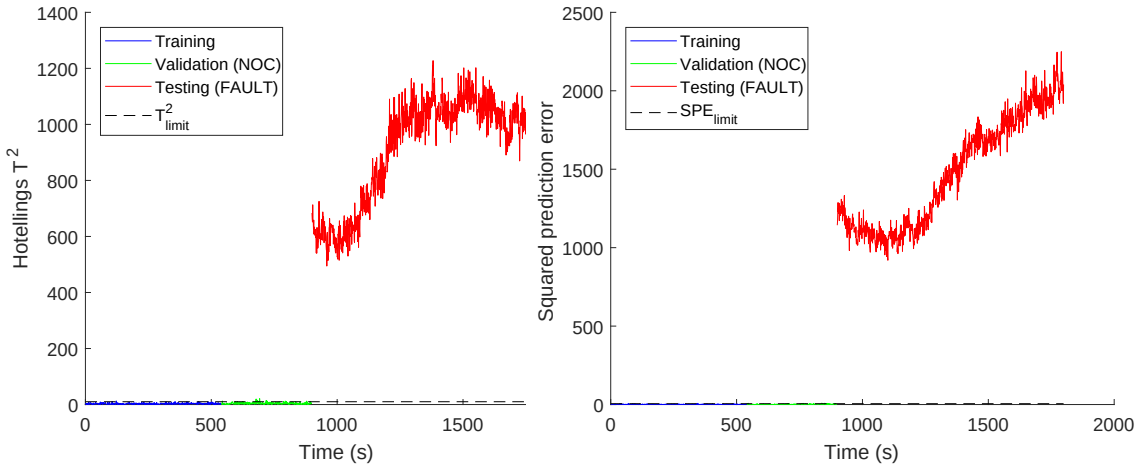


(c) Average relative contributions.

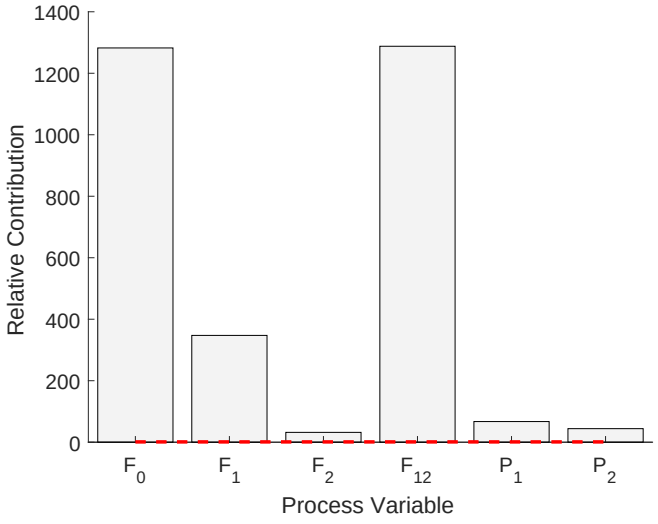
Figure E.8: MSPM approach fault diagnosis results: Configuration N two-tank system with P2s fault manifesting after 900 seconds.

E.3 Configuration NC

E.3.1 aConn fault



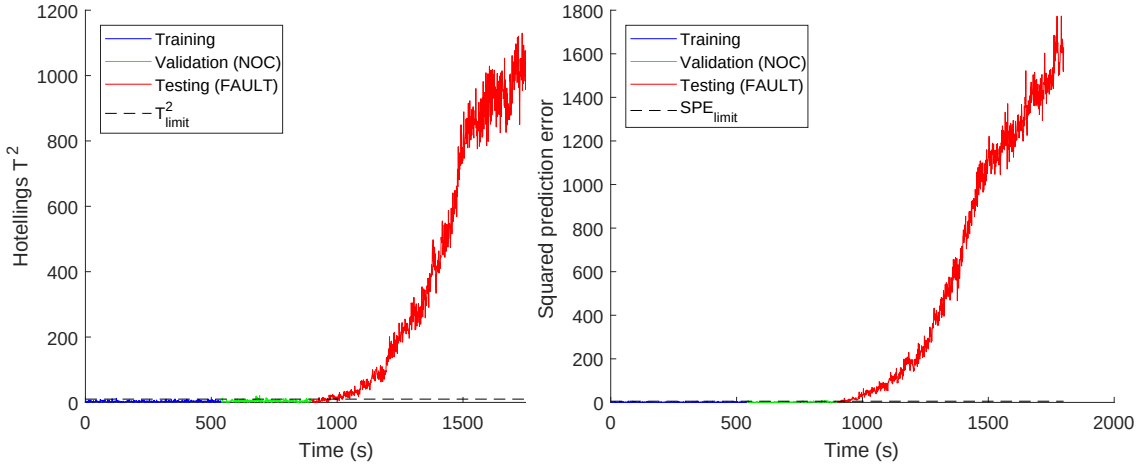
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



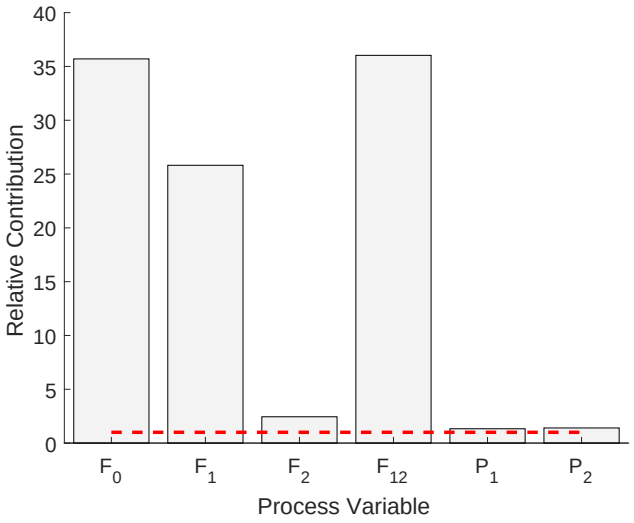
(c) Average relative contributions.

Figure E.9: MSPM approach fault diagnosis results: Configuration NC two-tank system with aConn fault manifesting after 900 seconds.

E.3.2 iConn fault



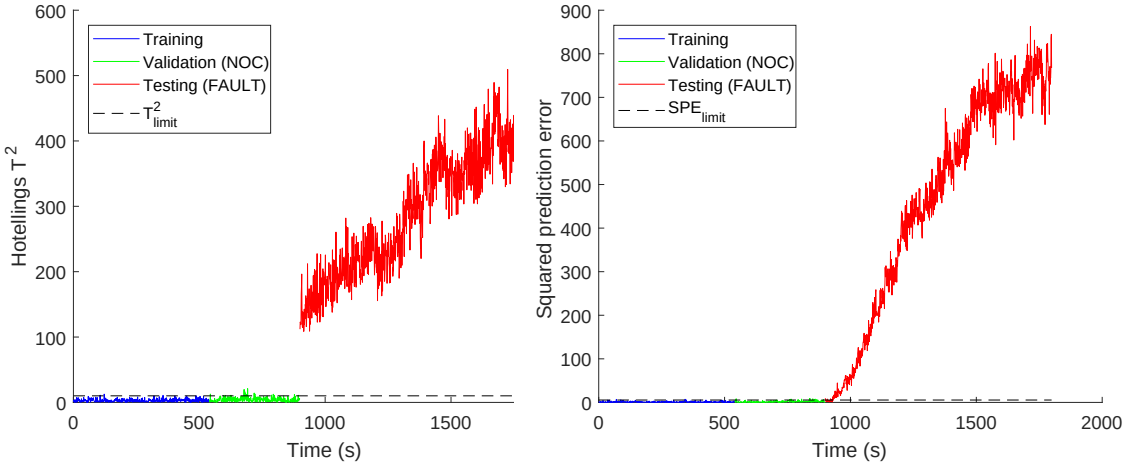
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



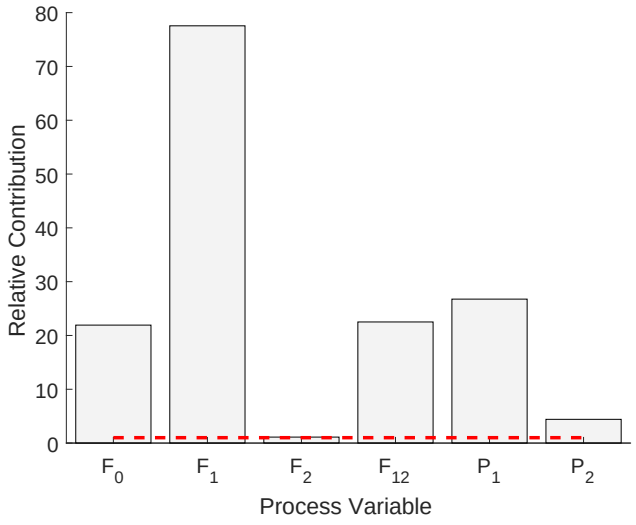
(c) Average relative contributions.

Figure E.10: MSPM approach fault diagnosis results: Configuration NC two-tank system with iConn fault manifesting after 900 seconds.

E.3.3 P1s fault



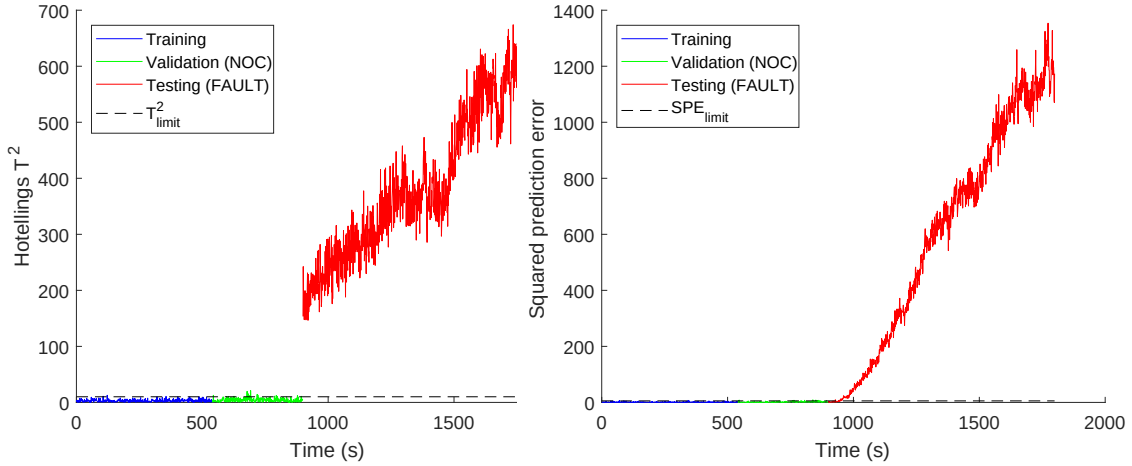
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



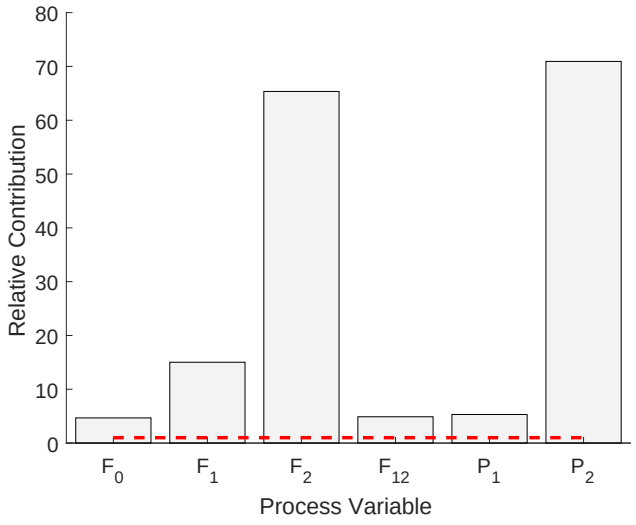
(c) Average relative contributions.

Figure E.11: MSPM approach fault diagnosis results: Configuration NC two-tank system with P1s fault manifesting after 900 seconds.

E.3.4 P2s fault



(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.

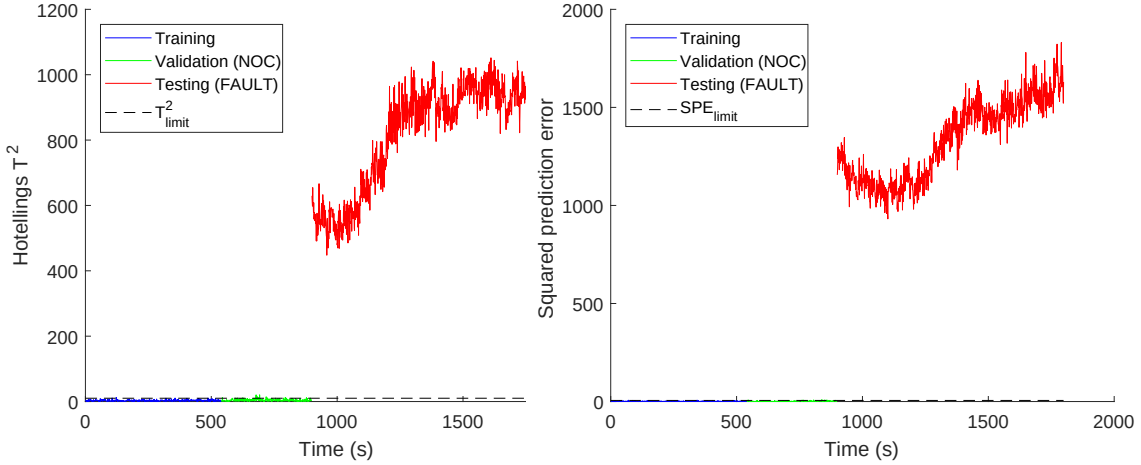


(c) Average relative contributions.

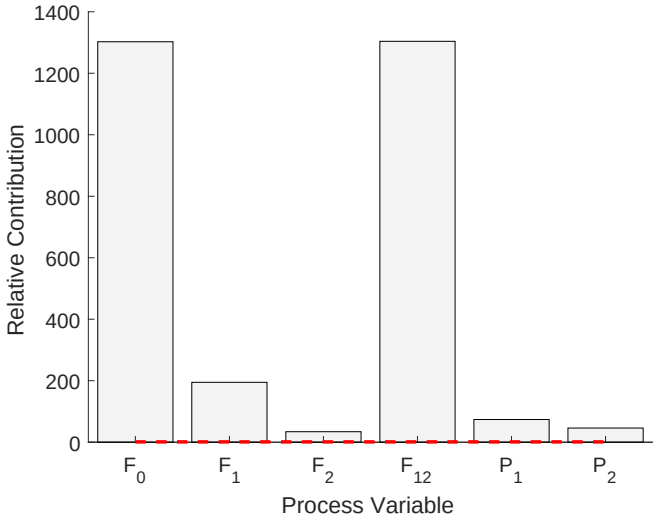
Figure E.12: MSPM approach fault diagnosis results: Configuration NC two-tank system with P2s fault manifesting after 900 seconds.

E.4 Configuration NCR

E.4.1 aConn fault



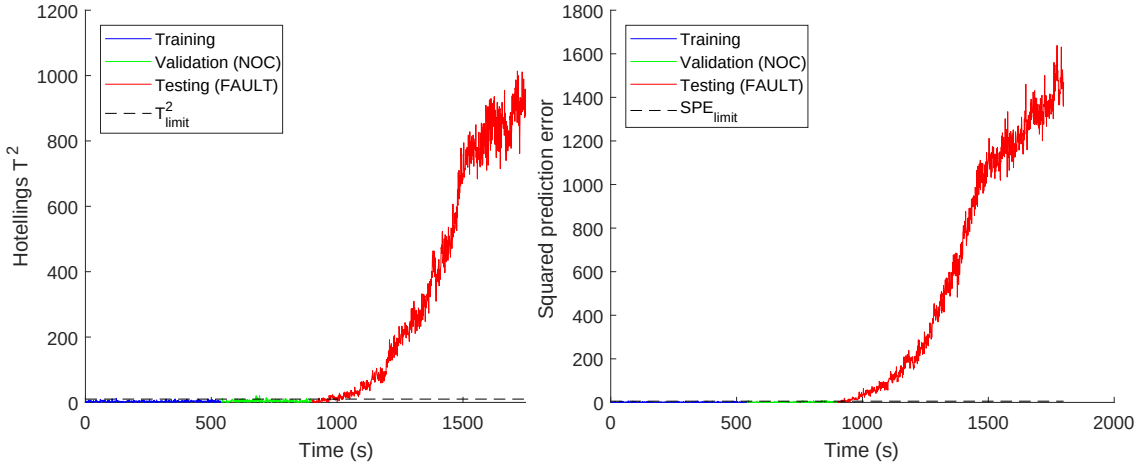
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



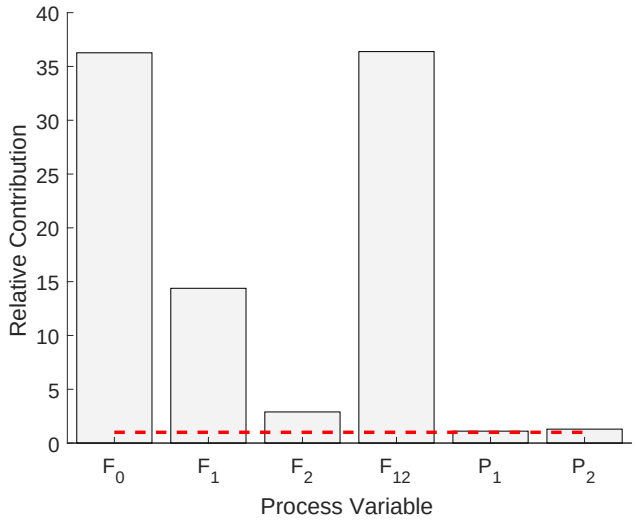
(c) Average relative contributions.

Figure E.13: MSPM approach fault diagnosis results: Configuration NCR two-tank system with aConn fault manifesting after 900 seconds.

E.4.2 iConn fault



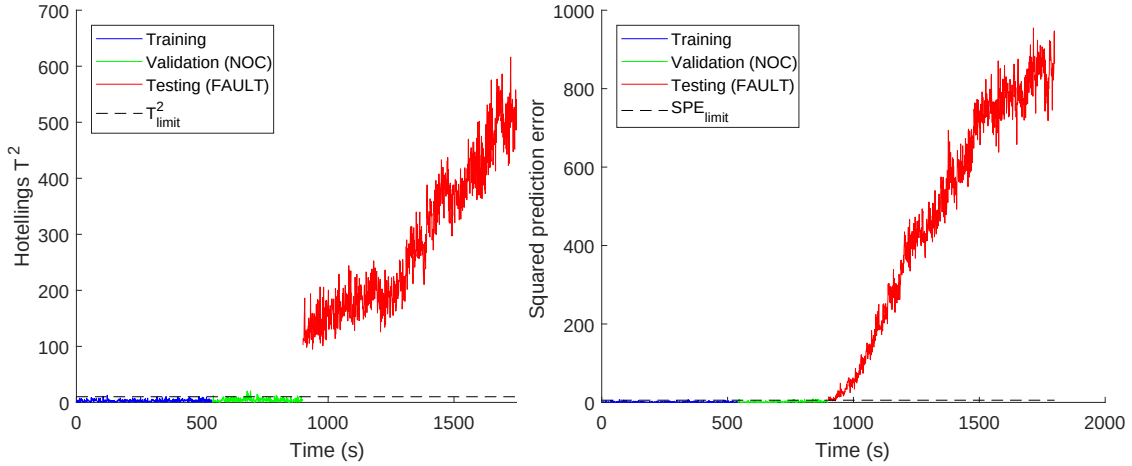
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



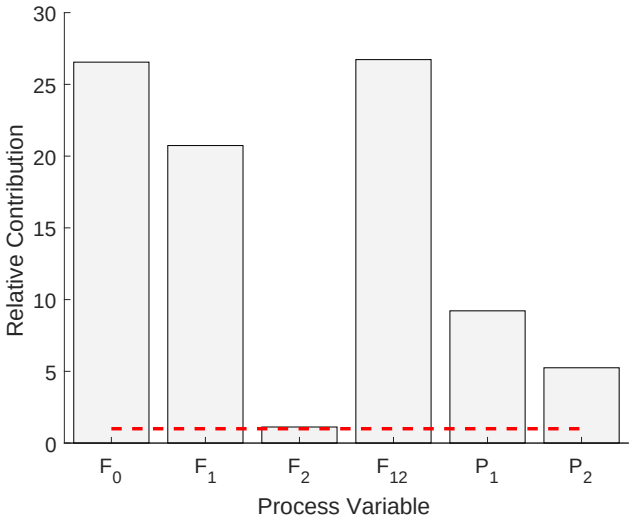
(c) Average relative contributions.

Figure E.14: MSPM approach fault diagnosis results: Configuration NCR two-tank system with iConn fault manifesting after 900 seconds.

E.4.3 P1s fault



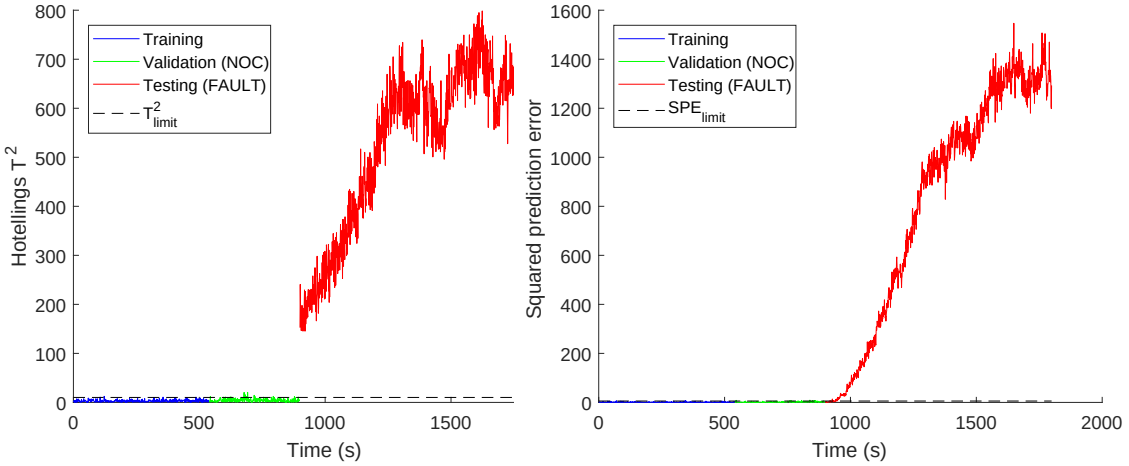
(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



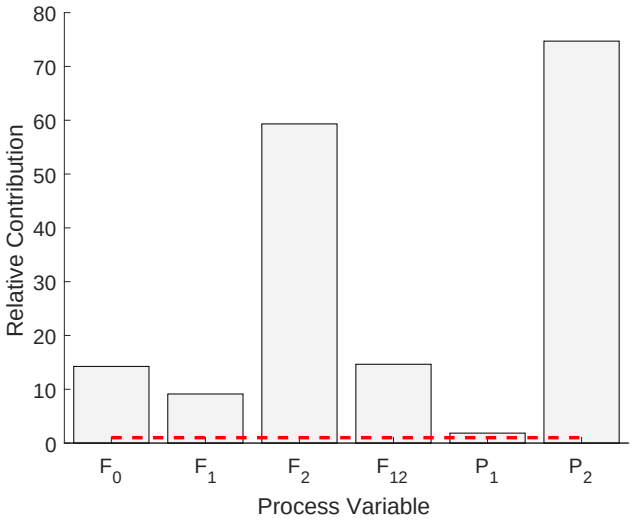
(c) Average relative contributions.

Figure E.15: MSPM approach fault diagnosis results: Configuration NCR two-tank system with P1s fault manifesting after 900 seconds.

E.4.4 P2s fault



(a) Modified Hotelling's T^2 statistic over time. (b) Squared prediction error statistic over time.



(c) Average relative contributions.

Figure E.16: MSPM approach fault diagnosis results: Configuration NCR two-tank system with P2s fault manifesting after 900 seconds.

Appendix F

Five-tank System: Simulated data

F.1 Configuration N

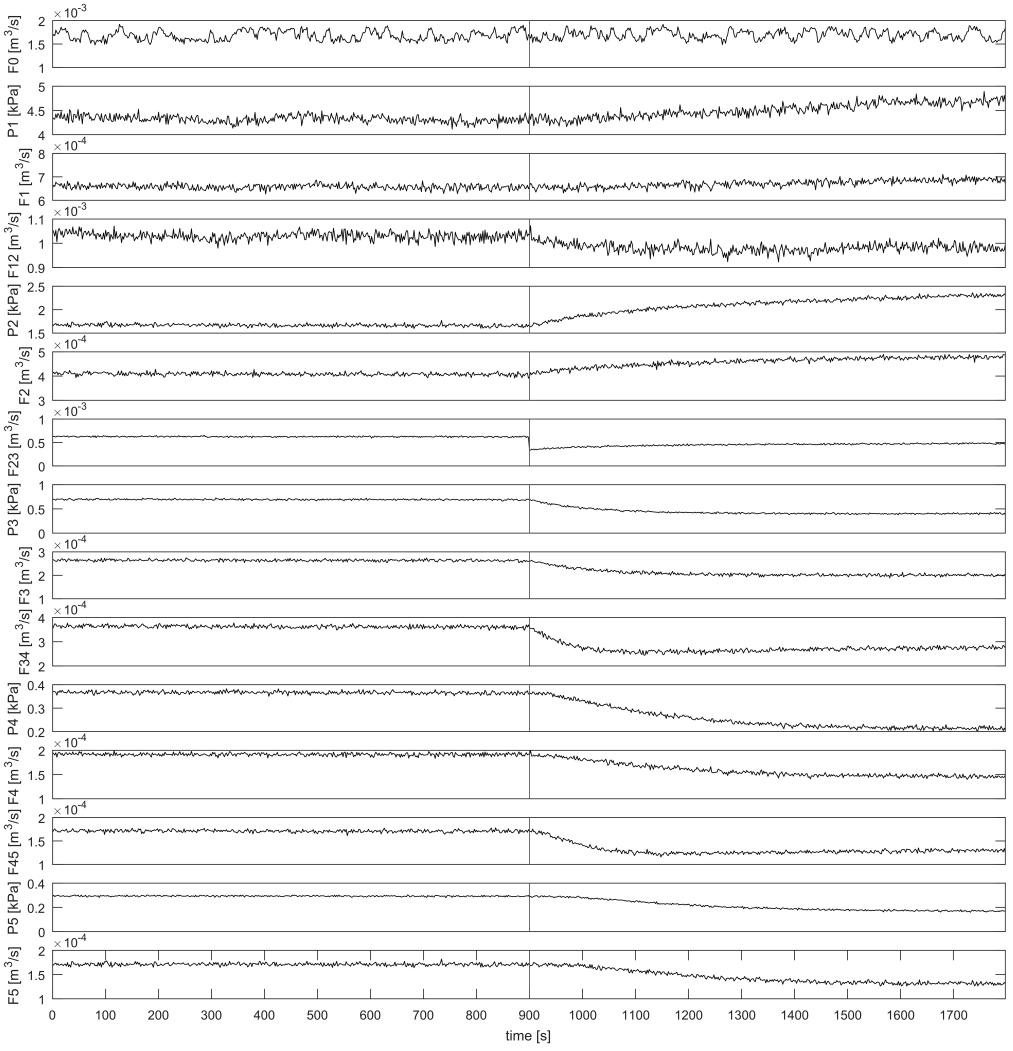


Figure F.1: Five-tank system process data: Configuration N with aConn type fault between tanks 2 and 3 manifesting after 900 seconds.

F.2 Configuration NC

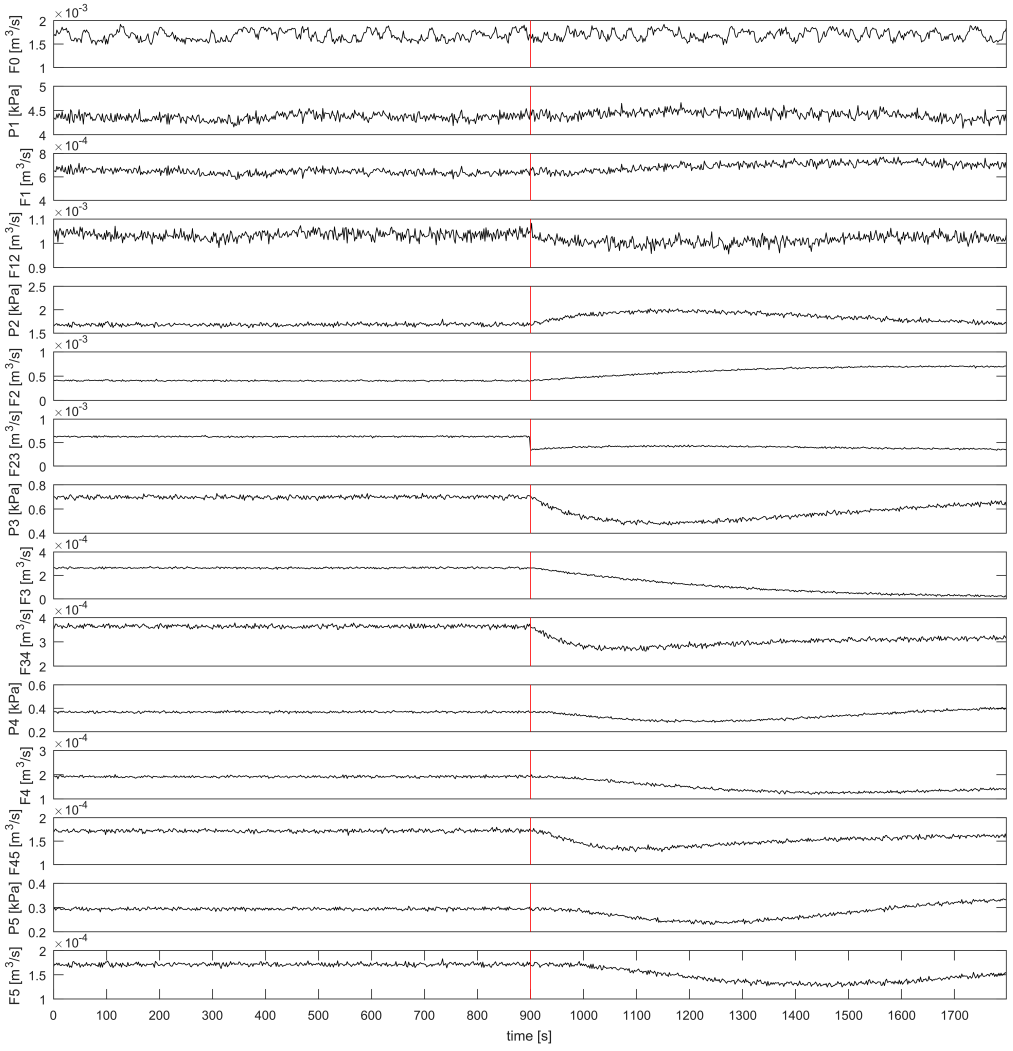


Figure F.2: Five-tank system process data: Configuration NC with aConn type fault between tanks 2 and 3 manifesting after 900 seconds.

F.3 Configuration NR

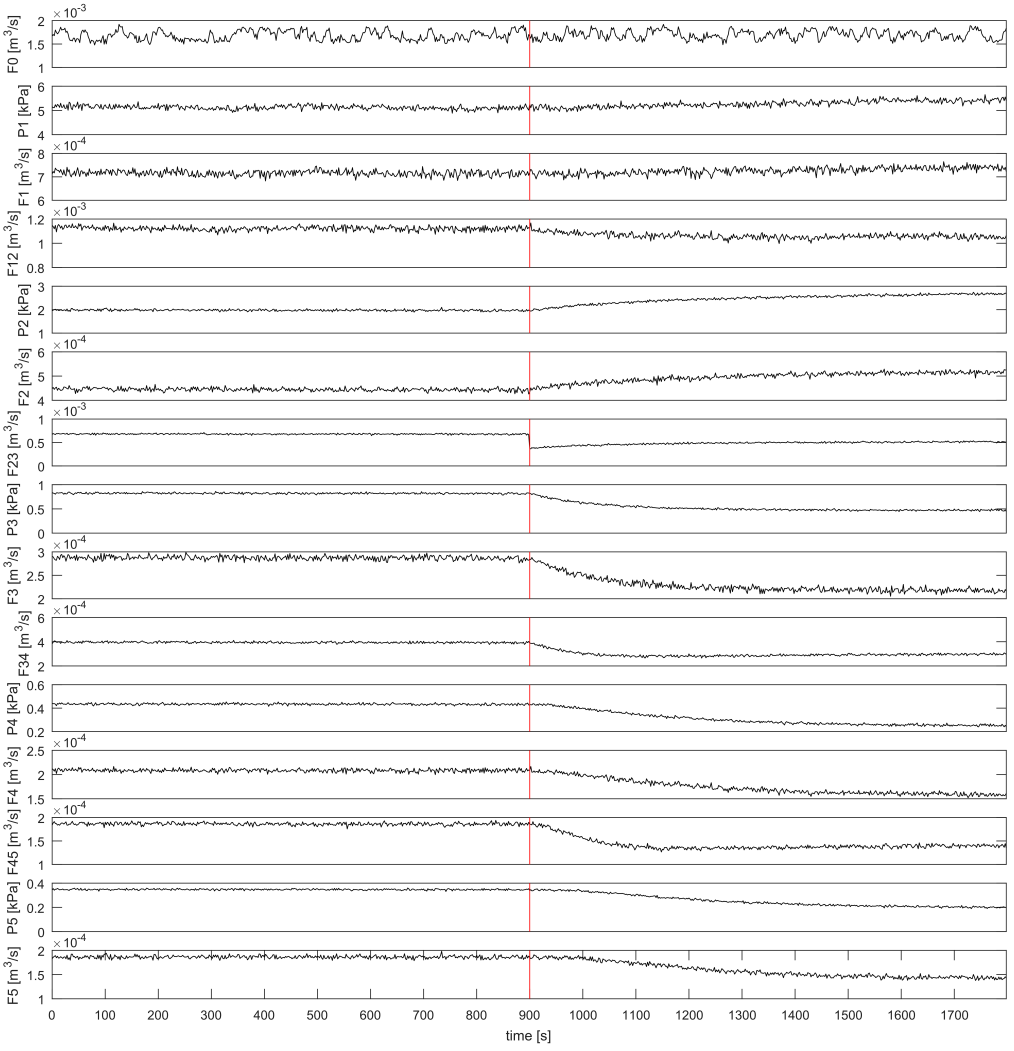


Figure F.3: Five-tank system process data: Configuration NR with aConn type fault between tanks 2 and 3 manifesting after 900 seconds.